



**8086
SOFTWARE TOOLBOX
VOLUME II**



.

.



.

.



**8086
SOFTWARE TOOLBOX
VOLUME II**

Order Number: 122310-001

Additional copies of this manual or other Intel literature may be obtained from:

Literature Department
Intel Corporation
3065 Bowers Avenue
Santa Clara, CA 95051

Intel retains the right to make changes to these specifications at any time, without notice. Contact your local sales office to obtain the latest specifications before placing your order.

Intel Corporation makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Intel Corporation assumes no responsibility for any errors that may appear in this document. Intel Corporation makes no commitment to update nor to keep current the information contained in this document.

Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in an Intel product. No other circuit patent licenses are implied.

Intel software products are copyrighted by and shall remain the property of Intel Corporation. Use, duplication or disclosure is subject to restrictions stated in Intel's software license, or as defined in ASPR 7-104.9(a)(9).

No part of this document may be copied or reproduced in any form or by any means without the prior written consent of Intel Corporation.

The following are trademarks of Intel Corporation and its affiliates and may only be used to identify Intel products:

BITBUS	i _m	iSBC	Plug-A-Bubble
COMMPuter	iMMX	iSBX	PROMPT
CREDIT	Insite	iSDM	Promware
Data Pipeline	int _{el}	iSXM	QueX
Genius	int _{el} BOS	KEPROM	QUEST
i	InteleVision	Library Manager	Ripplemode
i ²	int _{el} igent Identifier	MCS	RMX/80
i ² ICE	int _{el} igent Programming	Megachassis	RUPI
ICE	Intellec	MICROMAINFRAME	Seamless
iCS	Intellink	MULTIBUS	SOLO
iDBP	iOSP	MULTICHANNEL	SYSTEM 2000
iDIS	iPDS	MULTIMODULE	UPI
iLBX	iRMX		

REV.	REVISION HISTORY	DATE	APPD.
-001	Original issue.	12/84	B.N.



This manual provides a thorough description of Intel's PL/M-86 syntax checking editor (PSCAN) and instructions for its use. The manual is directed to PL/M-86 programmers with varying degrees of experience.

PSCAN contains the conventional text editing capabilities of Intel's AEDIT-86 interactive text editor, Version 1, which provide for entering, copying, deleting, and moving text within a file. The language oriented editing capabilities of PSCAN enable a programmer to perform interactive editing functions such as syntax checking, automatic formatting, and limited semantic processing of LITERALLYs. These capabilities increase efficiency in the program development cycle by reducing or eliminating the need for redundant compilations. Syntax errors that are often not detected until a program is compiled can be identified much earlier by PSCAN and corrected before the source program is compiled.

If you have used the AEDIT-86 text editor before, it may be unnecessary for you to read all of the command descriptions in Chapter 3. Most of the text editing commands function in the same way as those in AEDIT-86, Version 1.

PSCAN is designed to run on the Intellec Series III Microcomputer Development System under the ISIS II/RUN or ISIS III/RUN operating system, on the Intellec Series IV Microcomputer Development System under the iNDX operating system, and under the iRMX operating system. It requires at least 160 Kbytes of memory to run. PSCAN will also run on various other terminals. For system specific information about PSCAN invocation or terminal configuration, see the appropriate appendixes.

The manual is divided into five chapters and nine appendixes as follows:

Chapter 1 provides an introduction to PSCAN.

Chapter 2 provides essential information and a brief tutorial that will enable you to start using PSCAN.

Chapter 3 provides detailed information about PSCAN's modes and commands.

Chapter 4 provides information necessary for creating and using macros.

Chapter 5 provides further description and examples of PSCAN's language oriented editing capabilities.

Appendix A lists PSCAN commands, their formats and the uses of special-purpose keys, function keys, and control characters.

Appendix B lists error messages and their causes.

Appendix C lists the ASCII character set with hexadecimal values.

Appendix D provides instructions for using PSCAN on a Series III.

Appendix E provides instructions for using PSCAN on a Series IV.

Appendix F provides instructions for using PSCAN under the iRMX operating system.

Appendix G provides information for configuring PSCAN for specific terminals.

Appendix H describes noninteractive use of PSCAN.

Appendix I contains a printed version of the sample program used in the tutorial in Chapter 2.

Appendix I contains a printed version of the sample program used in the tutorial in Chapter 2.

A glossary and an index follow Appendix I.

NOTATIONAL CONVENTIONS

UPPERCASE	Characters shown in uppercase must be entered in the order shown. You may enter the characters in uppercase or lowercase.
filename	Is a valid name for a disk file.
system-id	Is a generic label placed on sample listings where an operating system-dependent name would actually be printed.
[]	Brackets indicate optional arguments or parameters.
{ }	One and only one of the enclosed entries must be selected unless the field is also surrounded by brackets, in which case it is optional.
punctuation	Punctuation other than ellipses, braces, and brackets must be entered as shown. For example, the punctuation shown in the following command must be entered:

SUBMIT PLM86(PROGA, SRC, '9 SEPT 84')

< > Indicates a special-purpose key or function key
 (for example, <cr>, or carriage return).

CONTROL Indicates a control character (the CONTROL key
 pressed with an alphanumeric key that performs a
 specific function).

RELATED PUBLICATIONS

The PSCAN Pocket Reference, order number 122195

Intellec® Series IV Microcomputer Development System Overview,
order number 121752

Intellec® Series IV Operating and Programming Guide, order number
121753

Intellec® Series III Programmer's Reference Manual, order number
121618

Intellec® Series III Microcomputer Development System Product
Overview, order number 121575

Introduction to the iRMX™ 86 Operating System, order number
983124

ISIS II User's Guide, order number 980306

iRMX™ 86 Operators Manual, order number 144523

Getting Started with iRMX™ 86 Systems Manual, order number 144349

iRMX™ 86 Configuration Guide, order number 983126

PL/M-86 Programming Manual, order number 980466

PL/M-86 User's Guide, order number 121636



8086 SOFTWARE TOOLBOX,
VOLUME II, V1.1

TABLE OF CONTENTS

CONTENTS	PAGE
CHAPTER 1	
INTRODUCTION TO PSCAN	
General Description.....	1-1
Capabilities.....	1-1
Screen Mode Editing.....	1-2
Configurability.....	1-3
CHAPTER 2	
USING PSCAN	
Introduction.....	2-1
Special-Purpose Keys, Control Characters, and Function Keys	2-2
Invocation Line.....	2-5
Tutorial.....	2-6
CHAPTER 3	
MODES AND COMMANDS	
Modes	3-1
Entering and Exiting Modes.....	3-1
Commands.....	3-2
Invoking Commands.....	3-2
Set.....	3-2
Find, -find.....	3-6
Insert.....	3-7
Replace, ?replace.....	3-8
Jump.....	3-9

BLOCK Mode Commands.....	3-10
Buffer.....	3-11
Delete.....	3-12
Get.....	3-13
Other.....	3-13
Tag.....	3-14
Undo.....	3-14
View.....	3-15
Again.....	3-15
Hex.....	3-15
Xchange.....	3-17
PL/M-86 ORIENTED COMMANDS.....	3-17
Move-Level-Out	3-18
Move-Level-In.....	3-18
Move-Statement-Up.....	3-19
Move-Statement-Down.....	3-19
Move-Token-Left.....	3-20
Move-Token-Right.....	3-20
QUIT Mode Commands.....	3-21
Abort.....	3-21
Exit.....	3-21
Init.....	3-21
Update.....	3-22
Write.....	3-22
Commands Available within Modes.....	3-24

Directional Arrow Keys and Delete Special-Purpose Keys.....	3-25
--	------

CHAPTER 4 MACRO MODE AND COMMANDS

Introduction.....	4-1
Macro Creation.....	4-1
Macro Names.....	4-2
Macro Files.....	4-2
Macro Examples.....	4-5
Macro Definitions.....	4-8
Set Commands in Macro Files.....	4-11
Macro Commands.....	4-11
Create.....	4-11
Get.....	4-13
Insert.....	4-13
List.....	4-14
Save.....	4-15
Execute Command.....	4-17
Using Macros in INSERT and XCHANGE Modes.....	4-17
Deleting Macros.....	4-18

CHAPTER 5 PSCAN FEATURES

Introduction.....	5-1
Syntax Checking.....	5-1
Automatic Formatting.....	5-3
Temporary Suspension of Automatic Formatting.....	5-4

How PSCAN Works	5-6
PL/M-86 Oriented Commands.....	5-7
APPENDIX A	
PSCAN COMMAND SUMMARY	
Special-Purpose Keys and Control Characters.....	A-6
Function Keys.....	A-9
APPENDIX B	
ERROR MESSAGES	
Editing Command Errors.....	B-1
Macro File Errors.....	B-3
Invocation Errors.....	B-4
APPENDIX C	
ASCII CODES	
APPENDIX D	
USING PSCAN ON THE SERIES III	
Memory.....	D-1
Invocation Command.....	D-1
Keyboard.....	D-1
APPENDIX E	
USING PSCAN ON THE SERIES IV	
Memory.....	E-1
Invocation Command.....	E-1
Keyboard.....	E-1

APPENDIX F USING PSCAN UNDER iRMX™

Memory.....	F-1
Invocation Command.....	F-1

APPENDIX G CONFIGURING PSCAN FOR A PARTICULAR TERMINAL

Configuration Commands.....	G-1
Configuring PSCAN.....	G-2

APPENDIX H NONINTERACTIVE USE

Noninteractive Input.....	H-1
Noninteractive Output.....	H-2

APPENDIX I CLOCK PROGRAM..... I-1

GLOSSARY

INDEX

FIGURES

FIGURES	TITLE	PAGE
2-1	PSCAN Display.....	2-1
2-2	Menu Prompt Lines.....	2-2
2-3	PSCAN Screen Display.....	2-7
2-4	Detected Syntax Error 1	2-9

2-5	Detected Syntax Error 2	2-11
2-6	Corrected Program.....	2-12
2-7	Corrected Program	2-13
2-8	Deleting Text.....	2-14
2-9	Program without Comments.....	2-15
3-1	Move-Level-In Command.....	3-18
3-2	Move-Statement-Down Command.....	3-19
3-3	Move-Token-Left Command.....	3-20
4-1	Macro Commands and Processes.....	4-4
4-2	Menu Prompt Line.....	4-12
5-1	Syntax Error Checking.....	5-2
5-2	FormattedCLOCK Program.....	5-3
5-3	Use of Special Comments.....	5-4
5-4	Changing Lines Affected by Special Comments.....	5-5
5-5	Move-Statement-Down Command.....	5-9
E-1	Series IV Keyboard.....	E-2

TABLES

TABLE	TITLE	PAGE
2-1	PSCAN's Use of Special-Purpose Keys and Control Characters	2-3
3-1	Commands Available within Modes.....	3-24
4-1	Macro Form Symbols.....	4-9
5-1	PL/M-86 Oriented Editing Commands.....	5-7
A-1	PSCAN Commands.....	A-1

A-2	PSCAN's Use of Special-Purpose Keys and Control Characters.....	A-2
C-1	ASCII Codes.....	C-1
G-1	Configuration Commands.....	G-4



GENERAL DESCRIPTION

PSCAN is an interactive, screen oriented editor with menu style command prompts. It is designed to be a tool for PL/M-86 programmers, providing support for the construction and manipulation of PL/M-86 program source files. The syntax checking feature of PSCAN can prevent redundant compile-edit cycles by identifying syntax errors in a program early in its development. With PSCAN, syntax errors can thus be corrected before the program is compiled.

PSCAN also offers an automatic formatting feature that consistently formats the PL/M-86 program as it is being entered.

Both the automatic formatting and the syntax checking features are optional; they can be turned off. With both of these features turned off, PSCAN can be used as a regular text editor. Automatic formatting is not available without syntax checking, however. This prevents a program containing syntax errors from being formatted.

PL/M-86 oriented editing commands are also available with PSCAN. These are commands that allow you to move around within your program using references to program structure as the guide for cursor movement. Because they rely on PL/M-86 program structure in order to function, these commands are most useful on program text that has been formatted. They are helpful for moving around quickly within a program during editing. Chapter 3 "Modes and Commands," and Chapter 5, "PSCAN Features," contain more information and examples of the PL/M-86 oriented editing commands.

PSCAN also does semantic processing of LITERALLYs. This offers you more programming flexibility because PSCAN understands LITERALLYs.

In addition to the above features, PSCAN contains all of the text editing commands necessary for entering, copying, deleting and manipulating text. These include commands for finding strings of characters, replacing them, deleting and storing blocks of text in a separate buffer, retrieving this text and jumping to the beginning or end of a file.

CAPABILITIES

In summary, PSCAN's capabilities allow you to -

- o Display and scroll text on the screen
- o Check your PL/M-86 source program for syntax errors as you enter it
- o Automatically format your program as you enter it
- o Move quickly through your formatted program from one statement, enclosed block or token to another
- o Move quickly to any position in a file or to any point on the screen
- o Correct typing mistakes as you type
- o Rewrite text by typing new letters or characters over old ones
- o Make insertions and deletions easily
- o Find any string of characters and substitute another string
- o Move or copy sections of text within a file or to another file
- o Create macros to execute several commands at once, simplifying repetitive tasks
- o Scan listing files while editing the main text file
- o Indent text automatically
- o View lines over 80 characters long

The text editing functions listed above should seem familiar to you if you have used Intel's AEDIT-86, Version 1. Essentially, PSCAN comprises the same text editing commands as AEDIT-86, Version 1, with the addition of the PL/M-86 oriented editing commands, the ability to perform syntax checking, and the ability to automatically format a program as completely new features.

SCREEN MODE EDITING

PSCAN takes advantage of the CRT display screen and keyboard to create a user-friendly environment. The screen acts as a window into the buffer where a copy of the text file to be edited is stored. The type of screen mode editing available through PSCAN allows you to see and verify changes to the text as they are made.

Most PSCAN commands function in a way that is directly related to the position of the cursor on the screen. For instance, to

delete a block of text you delimit the block by positioning the cursor at the start and at the end of the block. When you insert text it always appears in the space just before the current cursor position.

The use of menu style command prompts makes command selection easy. During an PSCAN session, the available commands are listed along the bottom of the display screen. Except for special-purpose keys, function keys and control characters that act as function keys, all available commands at the current mode of PSCAN are displayed at the bottom of the display screen. Special-purpose keys, function keys and control characters that have specific functions are described in Chapter 2.

CONFIGURABILITY

Another important aspect of the PSCAN is its configurability. It will configure itself to function correctly on the Series III or Series IV terminals by default. It is also possible to create a separate file containing configuration commands that enable PSCAN to run on a particular terminal. These configuration commands are used to inform the PSCAN program as to what the particular terminal looks like (number of lines on the screen, its delete characters, etc.).

Configuring PSCAN to run on particular terminals is described in more detail in Appendix G.

Appendix D describes running PSCAN on a Series III, Appendix E describes running PSCAN on a Series IV, and Appendix F describes running PSCAN under the iRMX operating system.



INTRODUCTION

This chapter provides essential information for using PSCAN and a brief tutorial. After reading this chapter and using the tutorial, you should know how to enter and exit PSCAN as well as what to expect during a PSCAN session.

Figure 2-1 shows what the screen display looks like after PSCAN is invoked. It shows you where the message line and prompt line appear as well as where the end of file (EOF) marker () and cursor first appear on the screen when PSCAN is invoked. The text area is where the text being edited appears. Status or error messages appear in the message line; commands or prompts that require input appear in the prompt line during PSCAN operation. When PSCAN is first invoked, it is at main command level, one of its modes of operation. Modes and commands are described in detail in Chapter 3.

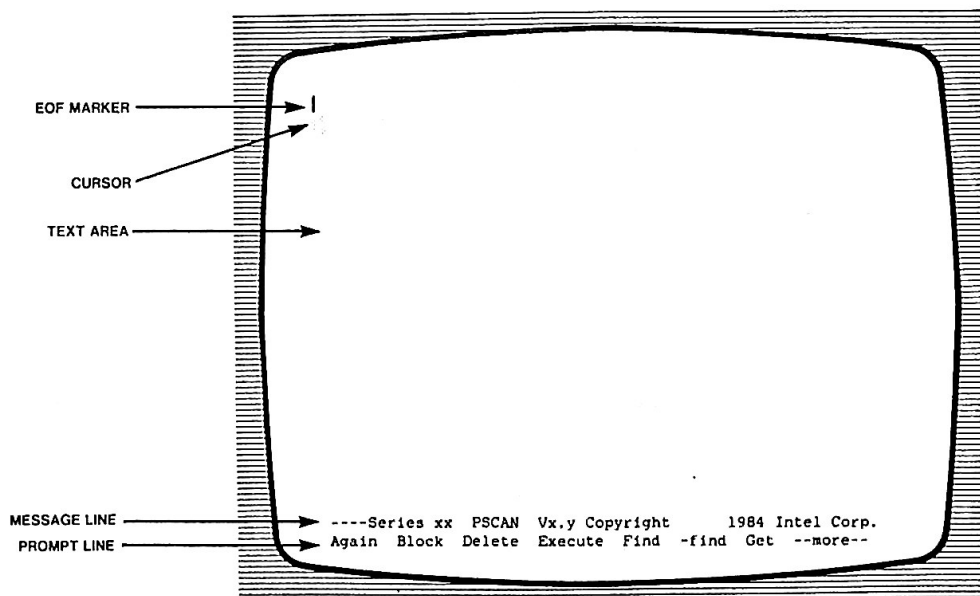


Figure 2-1. PSCAN Display

The commands that are available within the current PSCAN mode are displayed on the menu prompt line. Generally, to invoke one of the commands displayed at the bottom of the display screen, press the first character of the command. It is also possible to invoke commands by pressing the control character configured to perform the desired function. On a Series IV, commands can also be invoked by pressing the function key on the keyboard that appears directly under the desired command displayed in the menu prompt line on the screen.

Because many commands are available at main command level, they do not all fit on the prompt line at once. For modes in which not all the commands fit on the menu prompt line, you can respond to the "more" prompt by pressing <TAB>. The TAB key causes the additional commands available (if any) to be displayed. There are three lines of commands available at main command level. Figure 2-2 shows the three different command lines available for use at main command level:

```
-----  
Again Block Delete Execute Find -find Get      --more--
```

```
-----  
Hex Insert Jump Macros Other Quit Replace      --more--
```

```
-----  
?replace Set Tag View Xchange                  --more--
```

Figure 2-2. Menu Prompt Lines

SPECIAL-PURPOSE KEYS, CONTROL CHARACTERS, AND FUNCTION KEYS

Before beginning a PSCAN session, it is essential for you to know what the different special-purpose keys, function keys, and control characters are and how they work. Note, however, that the functions these keys perform are not because of the keys themselves but because of the way that the PSCAN program interprets the codes generated by the keys when they are pressed.

In this manual, the term special-purpose key refers to certain non-alphanumeric keys (keys that do not represent letters or numbers) that appear on the keyboard with a specific label. When pressed, special-purpose keys generate codes that normally cause PSCAN to perform specific functions such as deleting a character, no matter what terminal is being used. For example, <RUBOUT> is a special-purpose key.

The term control character refers to certain alphanumeric keys that, when pressed with the key labeled CONTROL (different terminals have different labels for CONTROL) cause PSCAN to perform specific functions. PSCAN can be configured to perform different functions after a particular control character is pressed. To determine what PSCAN does when a particular control character is pressed, see Appendix G.

The Series IV contains eight function keys labeled F0-F7 on its keyboard, in addition to its special-purpose keys and control characters. These function keys appear just under the display screen and are usually configured to correspond to the commands displayed in the menu prompt line under which they appear on the keyboard. Because the screen display changes and commands displayed in the menu prompt line change, the PSCAN command that a certain function key invokes also changes throughout a PSCAN session.

Table 2-1 lists the default usage of special-purpose keys for a Series III. A few Series IV-specific keys are also indicated in the table as well as the default usage for a few control characters.

Table 2-1. PSCAN's Use of Special-Purpose Keys and Control Characters

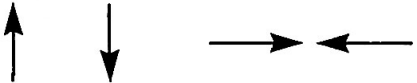
Special-Purpose Keys	Purpose
Directional Arrows  <RETURN> <TAB>	Move the cursor in the direction of the arrow. Places the cursor at the beginning of the next line of text. Indicated in this manual by <cr>. Causes the next line of available commands to be displayed in the menu prompt line of the display screen.

Table 2-1. PSCAN's Use of Special-Purpose Keys and Control Characters (Cont'd.)

<RUBOUT>	Deletes the character that precedes the cursor. The cursor moves back one, taking the place of the deleted character.
<HOME>	Pressed with an arrow key, moves the cursor continuously in the direction of the arrow (intensifies the action of the arrow key).
<ESC>	Returns PSCAN to main command level, from which all other modes can be entered. This key does not return PSCAN to main command level from MACRO mode, however.
<TPWR>	Controls the case of letter characters.
<CAPS LOCK> (Series IV)	Controls the case of letter characters on a Series IV.
<DEL CHAR> (Series IV)	Deletes the character under the cursor (on a Series IV).
<DELETE LINE> (Series IV)	Deletes the line in which the cursor is currently positioned.
Control Characters	Purpose
CONTROL A	Deletes the character under the cursor and all characters to the right of the cursor in the line in which the cursor is positioned.
CONTROL C	Aborts the most recently invoked command and returns PSCAN to main command level.
CONTROL D	Traps to the debugger.

Table 2-1. PSCAN's Use of Special-Purpose Keys
and Control Characters (Cont'd.)

CONTROL F	Deletes the character under the cursor.
CONTROL U	Restores characters deleted by previous delete command.
CONTROL X	Deletes all characters to the left of the cursor in the line in which the cursor is positioned.
CONTROL Z	Deletes the line in which the cursor is positioned.

INVOCATION LINE

The following command line invokes PSCAN on a Series III:

```
RUN PSCAN [input file [TO output file] [, other input file [TO
other output file] [MACRO (filename) | NOMACRO]
```

On a Series IV, the invocation line is the same except that the RUN parameter is not necessary:

```
PSCAN [input file [To output file] [, other input file
[TO other output file] [MACRO (filename) | NOMACRO]
```

Each of the different files is explained below.

Input file This is the name of the file you plan to edit. If this filename is not specified at invocation, PSCAN allows you to enter text in its edit buffer. If you wish to save this text, you must supply a filename (for which you are prompted by PSCAN) when you exit.

Output file PSCAN offers you the option of naming an output file to which the text contained in the edit buffer is written when you exit the PSCAN (using the QUIT Exit command or the QUIT Update command). This can be a new file or an existing file. When you write to an existing file, the old file is overwritten. Note that to write to an existing file, ensure that you have write permission on that file.

	See Chapter 3 for a detailed discussion of the QUIT mode commands.
Other input file	This file's contents will be copied to PSCAN's secondary buffer called the OTHER buffer. See Chapter 3 for a description of the Other command.
Other output file	This is the destination file to which text from the OTHER buffer is written when you exit PSCAN. See Chapter 3 for a description of the Other command.
MACRO	This changes the macro file for which PSCAN searches. (Normally it searches for the default macro file named PSCAN.MAC, but by entering MACRO and a filename, you can specify the MACRO file you wish PSCAN to use.) The abbreviation for MACRO, MR can also be used, followed by the filename. See Chapter 4 for a discussion on MACRO mode and commands.
(filename)	This is the name of the file that PSCAN should use instead of the default macro file (PSCAN.MAC). This filename must be typed within parentheses.
NOMACRO	This indicates to PSCAN that you do not want it to read any macro file upon invocation. You can also enter its abbreviation, NOMR.

TUTORIAL

This section consists of a brief tutorial in which you will practice invoking, using and exiting the PSCAN. Not all of the available commands are presented in this tutorial. Its purpose is to familiarize you with PSCAN and to prepare you for its use.

In order to best use this tutorial, perform the steps as they are described, in conjunction with the sample file named CLOCK.SRC, which is included in your release diskette. This file contains a short, sample PL/M-86 program named CLOCK. The sample program contains a few errors that you will correct during the following tutorial.

Appendix I contains a printed version of the CLOCK program used in this tutorial.

Before Starting PSCAN

- o Refer to the appendix that describes running PSCAN on the specific system you are using.

- o If you are using anything other than a Series III or a Series IV, ensure that your directory contains a configuration file for your specific terminal. Either copy this file into the file named PSCAN.MAC (the default MACRO file for which PSCAN automatically searches upon invocation) or remember the name of the file and enter its name in the invocation line.

PSCAN's syntax checking and automatic formatting features are enabled during this tutorial because that is their default condition.

Step 1

Enter the appropriate invocation line for your machine:

PSCAN CLOCK.SRC TO TEMP

If you are using a Series III, begin the invocation line shown above with RUN:

RUN PSCAN CLOCK.SRC TO TEMP

You are entering TEMP as the output file because you may want to go back and reuse the CLOCK.SRC file to practice PSCAN on your own. The changes you make during this tutorial will be written to the TEMP file.

The screen shown in Figure 2-3 appears after PSCAN is invoked.

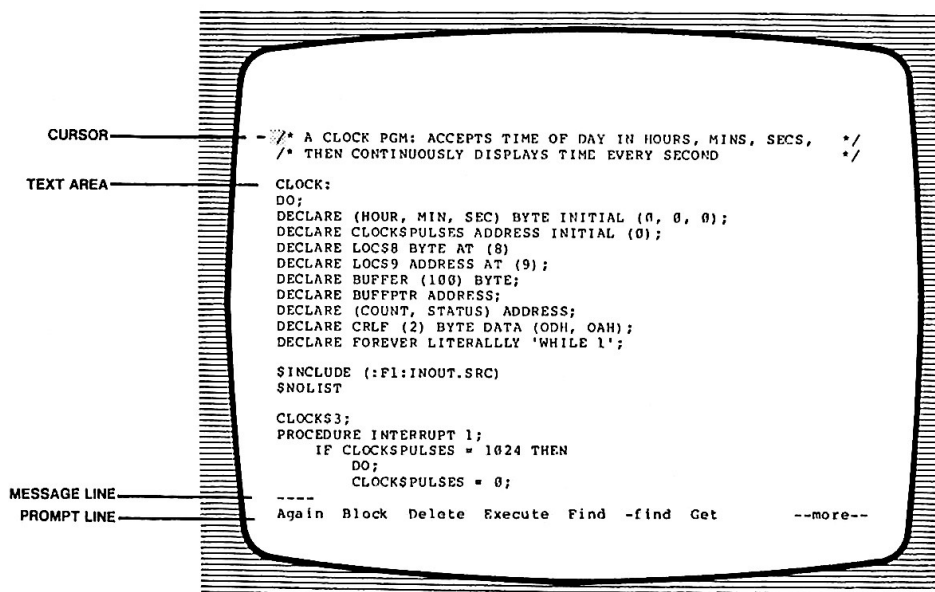


Figure 2-3. PSCAN Screen Display

The program contained in the file named CLOCK is read into the edit buffer, which is displayed in the text area of the screen.

You are now at main command level. PSCAN has six modes of operation, each of which is described in the next chapter.

Step 2

Press the down arrow key
and hold it down.

(This is to examine the
program, line by line, as the
cursor moves down the
program.)

The following message appears:

----formatting

(This indicates that PSCAN is
starting to format the program
with its automatic formatting
feature.)

Next, another message appears:

----syntax error

(PSCAN has detected a syntax
error in the program. It
stops further syntax checking
and formatting.)

Step 3

Press J.

(to invoke the Jump command)

The following menu prompt line appears:

Start End eRror Line Position --more--

(These are the subcommands available
for the Jump command.)

Step 4

Press R.

(This makes the cursor "jump" to the location of the syntax error.)

The cursor moves to the line containing the syntax error or to a line near the line containing the syntax error, as shown in Figure 2-4.

```

/* A CLOCK PGM: ACCEPTS TIME OF DAY IN HOURS, MINS, SECS, */
/* THEN CONTINUOUSLY DISPLAYS TIME EVERY SECOND */
CLOCK:
DO;
DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0, 0, 0);
DECLARE CLOCKSPULSES ADDRESS INITIAL (0);
DECLARE LOCS8 BYTE AT (8);
DECLARE LOCS9 ADDRESS AT (9);
DECLARE BUFFER (100) BYTE;
DECLARE BUFFPTR ADDRESS;
DECLARE (COUNT, STATUS) ADDRESS;
DECLARE CRLF (2) BYTE DATA (ODH, OAH);
DECLARE FOREVER LITERALLY 'WHILE 1';

SINCLUDE (:F1:INOUT.SRC)
SNOLIST

CLOCK$3;
PROCEDURE INTERRUPT 1;
  IF CLOCKSPULSES = 1024 THEN
    DO;
      CLOCKSPULSES = 0;
    ----
  Again Block Delete Execute Find -find Get      --more--

```

Figure 2-4. Detected Syntax Error 1

Step 5

Using the necessary arrow key(s),
move the cursor to the space
just after the line ending with -

(8)

(This line must end with a
semicolon for correct syntax.)

Step 6

Press I.

(This is to enter the INSERT
mode.)

The following message appears to indicate that you
are in the INSERT mode:

[insert]

Step 7

Enter ;.

(This corrects the syntax
error.)

Step 8

Press <ESC>.

(This returns PSCAN to main
command level.)

Step 9

Press the down arrow key
and hold it down.

(This resumes examination of the
program, line by line. PSCAN
continues to format.)

Step 10

The following message appears:

----syntax error

(PCAN has detected another
syntax error.)

Step 11

Press J.

(This invokes the Jump command.)

The following menu prompt line appears:

```
----
Start End eRror Line Position --more--
```

Step 12

Press R.

This makes the cursor "jump" to the line containing the syntax error.)

The cursor moves to the line containing the syntax error or to a line near the line containing the syntax error, as shown in Figure 2-5.

```

/* A CLOCK PGM: ACCEPTS TIME OF DAY IN HOURS, MINS, SECS. */
/* THEN CONTINUOUSLY DISPLAYS TIME EVERY SECOND */
CLOCK:
DO;
DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0, 0, 0);
DECLARE CLOCKSPULSES ADDRESS INITIAL (0);
DECLARE LOCS8 BYTE AT (8);
DECLARE LOCS9 ADDRESS AT (9);
DECLARE BUFFER (100) BYTE;
DECLARE BUFFPTR ADDRESS;
DECLARE (COUNT, STATUS) ADDRESS;
DECLARE CRIF (2) BYTE DATA (0DH, 0AH);
DECLARE FOREVER LITERALLY 'WHILE 1';

$INCLUDE (:P1:INOUT.SRC)
$NOLIST

CLOCK$3;
PROCEDURE INTERRUPT 1;
  IF CLOCKSPULSES = 1024 THEN
    DO;
      CLOCKSPULSES = 0;
    ----
  Again Block Delete Execute Find -find Get  --more--

```

Figure 2-5. Detected Syntax Error 2

Step 13

Using the necessary arrow key,
move the cursor to the third "L"
in the word -

LITERALLLY

(It is misspelled.)

Step 14

Press <RUBOUT>.

The extra "L" is deleted as shown in Figure 2-6.

```

/* A CLOCK PGM: ACCEPTS TIME OF DAY IN HOURS, MINS, SECS, */
/* THEN CONTINUOUSLY DISPLAYS TIME EVERY SECOND */
CLOCK:
DO;
DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0, 0, 0);
DECLARE CLOCKSPULSES ADDRESS INITIAL (0);
DECLARE LOCS8 BYTE AT (8);
DECLARE LOCS9 ADDRESS AT (9);
DECLARE BUFFER (100) BYTE;
DECLARE BUFFPTR ADDRESS;
DECLAPE (COUNT, STATUS) ADDRESS;
DECLARE CRLF (2) BYTE DATA (0DH, 0AH);
DECLARE FOREVER LITERALLY 'WHILE 1';

$INCLUDE (:F1:INOUT.SRC)
$NOLIST

CLOCK$3;
PROCEDURE INTERRUPT 1;
  IF CLOCKSPULSES = 1024 THEN
    DO;
      CLOCKSPULSES = 0;
    ----
  Again Block Delete Execute Find -find Get      --more--

```

Figure 2-6. Corrected Program

Step 15

Press <ESC>.

(This returns PSCAN to main command level.)

Step 16

Press J.

(This invokes the Jump command.)

The following menu prompt line appears:

— — — — —

Start	End	eRror	Line	Position	--more--
-------	-----	-------	------	----------	----------

Step 17

Press S.

(This moves the cursor to the beginning of the file.)

The cursor moves to the beginning of the file, as illustrated in Figure 2-7.

```

/** A CLOCK PGM ACCEPTS TIME OF DAY IN HOURS, MINS, SECS,
/** THEN CONTINUOUSLY DISPLAYS TIME EVERY SECOND
/**/

CLOCK:
DO;
DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0, 0, 0);
DECLARE CLOCKSPULSES ADDRESS INITIAL (0);
DECLARE LOC$8 BYTE AT (8);
DECLARE LOC$9 ADDRESS AT (9);
DECLARE BUFFER (100) BYTE;
DECLARE BUFFPTR ADDRESS;
DECLARE (COUNT, STATUS) ADDRESS;
DECLARE CRLF (2) BYTE DATA (0DH, 0AH);
DECLARE FOREVER LITERALLY 'WHILE 1';

$INCLUDE (:FI:INOUT.SRC)
$NOLIST

CLOCK$3;
PROCEDURE INTERRUPT 1;
    IF CLOCKSPULSES = 1024 THEN
        DO;
            CLOCKSPULSES = 0;
        ;
    ;
END;

-----
Again Block Delete Execute Find -find Get          --more--

```

Figure 2-7. Corrected Program

To practice deleting text, you are going to delete the comments at the beginning of the file.

Step 18

Ensure that the cursor is located over the first character of the first comment (the slash before the word A).

Press D to invoke the BLOCK Delete command.

(To delete the comments, you must first delimit the text to be deleted.)

An @ replaces the first slash character (where the cursor was positioned) and a new menu prompt line appears at the bottom of the display screen shown in Figure 2-8.

```

@* A CLOCK PGM: ACCEPTS TIME OF DAY IN HOURS, MINS, SECS,  */
/* THEN CONTINUOUSLY DISPLAYS TIME EVERY SECOND          */
CLOCK:
DO;
DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0, 0, 0);
DECLARE CLOCKSPULSES ADDRESS INITIAL (0);
DECLARE LOCS8 BYTE AT (8);
DECLARE LOCS9 ADDRESS AT (9);
DECLARE BUFFER (100) BYTE;
DECLARE BUFPTR ADDRESS;
DECLARE (COUNT, STATUS) ADDRESS;
DECLARE CRLF (2) BYTE DATA (0DH, 0AH);
DECLARE FOREVER LITERALLY 'WHILE 1';

$INCLUDE (:F1:INOUT.SRC)
$NOLIST

CLOCK$3;
PROCEDURE INTERRUPT 1;
  IF CLOCKSPULSES = 1924 THEN
    DO;
      CLOCKSPULSES = 0;
    ----
  Buffer Delete Find -find Jump Put

```

Figure 2-8. Deleting Text

Step 19

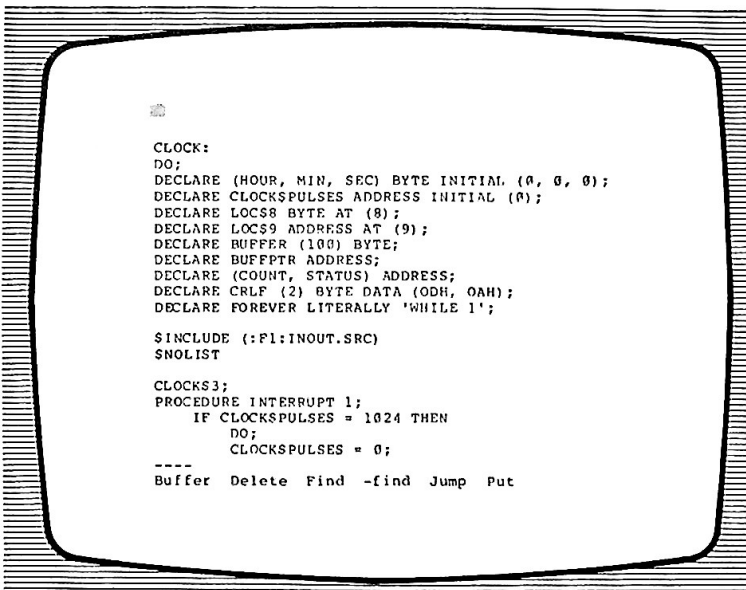
Move the cursor to the space immediately after the last character (the slash) of the second comment.

Step 20

Press D.

(This indicates the end of the text to be deleted.)

The two comment lines are deleted, as illustrated in Figure 2-9.



```

CLOCK:
DO;
DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0, 0, 0);
DECLARE CLOCKSPULSES ADDRESS INITIAL (0);
DECLARE LOC88 BYTE AT (8);
DECLARE LOC89 ADDRESS AT (9);
DECLARE BUFFER (100) BYTE;
DECLARE BUFFPTR ADDRESS;
DECLARE (COUNT, STATUS) ADDRESS;
DECLARE CRLF (2) BYTE DATA (0DH, 0AH);
DECLARE FOREVER LITERALLY 'WHILE 1';

$INCLUDE (:F1:INOUT.SRC)
$NOLIST

CLOCK$3;
PROCEDURE INTERRUPT 1;
  IF CLOCKSPULSES = 1024 THEN
    DO;
      CLOCKSPULSES = 0;
    ----
  Buffer Delete Find -find Jump Put
  
```

Figure 2-9. Program without Comments

You are now ready to finish this tutorial by exiting PSCAN. Because you have made corrections to the CLOCK program, you probably want to save them. To exit PSCAN saving the changes you have made:

Step 21

Press Q.

(This enters the QUIT mode.)

The following menu prompt line appears:

```
----Editing CLOCK.SRC TO TEMP
Abort  Exit  Init  Update  Write
```

Step 22

Press E.

(This invokes the Exit command.)

The following message appears:

```
----TEMP has been written
```

This message indicates that the changes you made to the CLOCK.SRC file during this session are saved in the TEMP file that you identified as the output file in the invocation line.

You are returned to the operating system level.

In this short tutorial you practiced inserting, deleting, and changing text. You also practiced using the Jump command and saw briefly how the syntax checking and automatic formatting features work.

The next chapter contains more detailed information about each of the modes and commands available in PSCAN. You may want to practice these on the sample program before using PSCAN on your own program.

Again, the CLOCK program is printed in Appendix I for your reference.

MODES

The functional behavior of PSCAN can be described in terms of its modes and commands. PSCAN has six modes of operation, from each of which it accepts a limited set of commands. These modes, or "states," are described below. All of the commands available within PSCAN (except Macro commands) are described in this chapter. MACRO mode and commands are discussed in detail in the next chapter.

- o Main command level is the mode in which PSCAN comes up when it is invoked.
- o INSERT mode is the mode in which you insert text at the current cursor position into the edit buffer.
- o XCHANGE mode is the mode in which you can replace old characters in the edit buffer with new ones.
- o QUIT mode accepts commands that allow you to exit the editor (saving the changes you have made), to save the file and continue editing, to stop editing the file without saving the changes you have made, and to start editing a new file.
- o BLOCK mode enables you to delimit blocks of your program or text for deletion and/or storage in the BLOCK buffer or in a specified file.
- o MACRO mode enables you to create and execute macros. MACRO mode and commands are described in detail in Chapter 4.

Entering and Exiting Modes

When PSCAN is invoked, it is at main command level. To enter any other mode you must first ensure that you are at main command level, and then press the first letter of the mode you wish to enter.

To exit the INSERT, BLOCK, or XCHANGE mode, press <ESC>. This returns you to the main command level, from which you use one of the QUIT mode commands to exit PSCAN or to change from the file you have been editing to a different file.

Commands

Many of the conventional text editing commands that work within the above modes are the same as those in version 1 of the AEDIT-86 interactive text editor. These commands control inserting, editing, and deleting text.

PSCAN's distinct power as a PL/M-86 syntax checking editor comes from its commands that allow you to turn the syntax checking or automatic formatting options on or off and from commands that enable you to move the cursor around (using references to program text) within the program being edited from one statement or token to another.

Some commands accept a repetition count. These commands allow you to enter a number for the number of times you want the command executed when you invoke the command. Relevant count information is included in the command explanations that follow.

To quickly determine what commands work in what modes, refer to Table 3-1 at the end of this chapter.

INVOKING COMMANDS

Before invoking a command, ensure that you are in the mode you wish to be in and that the mode accepts the specific command you plan to invoke.

Invoking a command is done by pressing the first letter of the command you wish to use.

If the command you invoke contains subcommands, they appear on the prompt line of the screen after you invoke the command. Subcommands are invoked in the same way that commands are invoked.

In addition, some commands require keyboard input. In these cases, after a command has been invoked, PSCAN prompts for the input it needs in order to complete the command.

To abort a command completely, press CONTROL C. This cancels the command and returns you to main command level.

Set

The Set command is available at main command level and in the MACRO mode. It is sometimes the first command to be used because it allows you to determine whether PSCAN will be used for syntax checking and automatic formatting of your program. It also enables you to set other options that affect how PSCAN operates during a particular editing session.

If the automatic formatting option is turned on, the syntax checking option is also on. It is possible to turn the syntax checking on and the automatic formatting option off, or to turn both of these options off. The default for both of these options is on.

Each of the subcommands within the Set command are described below. The prompt that appears on the screen when the particular subcommand is invoked is also shown. For prompts with yes or no prompts, the default is bracketed. If you change the default, the option you choose becomes bracketed.

- o Autocr - permits automatic insertion of a carriage return/linefeed.

insert cr,lf automatically? (y or [n])

If you enter y, a carriage return/line feed is inserted in the last column on the screen whenever an attempt is made to insert a character in that column. Trailing blanks and tabs are deleted and the carriage return/line feed is inserted between words or after a special character.

If you enter n, the option is turned off.

- o BAK - permits the automatic creation of a backup file.

create .BAK files? ([y] or n)

If you enter y, the file you are editing is renamed file.BAK before the QUIT Exit or QUIT Update command replaces the original file with the contents of the edit buffer.

If you enter n, the option is turned off.

CAUTION

Turning off this option can be dangerous because the backup file protects against accidental loss or damage of your file.

- o Case - determines whether case is important in finding strings. (If you indicate that case is not important, during a search for a target string, PSCAN will not consider whether the string appears in upper or lowercase.)

ignore case of Find target? ([y] or n)

- o Err_check- permits turning the syntax checking option on or off.

set syntax error checking? ([y] or n)

- o Format - permits turning the automatic formatting option on or off.

set automatic formatting? ([y] or n)

NOTE

Automatic formatting can be temporarily suspended during an editing session by inserting the comment `/*-f*/` in the line preceding the lines of code you wish to remain unformatted (you may wish to format these lines yourself).

The comment `/*+f*/` must be inserted before automatic formatting will restart. These comments work only when the automatic formatting option is on.

- o Indent - permits selection of automatic indenting

automatically indent during insertion? (y or [n])

If you enter y, inserted carriage returns are followed by the same combination of blanks and tabs as the preceding line. Program text lines are indented, using tabs, to reflect the structure of the program.

If you enter n, the option is turned off.

- o Leftcol - permits viewing of lines over 80 characters long (default is zero).

first column:

This subcommand allows you to view lines over 80 characters long on the screen and accepts any number from 0 to 80 (columns start at zero). The number input indicates the number of characters at the start of a line that should not be displayed.

For example, if a line is 90 characters long, set the left column to 10 and the screen will display the line from column 1 to 89.

An exclamation point (!) is printed in column 0 if any characters are not displayed.

- o Notab - permits insertion of blanks for tabs.

insert blanks for tabs? (y or [n])

If y, blanks are inserted instead of tabs whenever you press <TAB> while in INSERT or XCHANGE mode.

If n, the option is turned off.

- o Showfind - permits a listing of all lines of text in which a string is found or replaced.

list lines on multiple finds? (y or [n])

If y, when a Find or Replace command is executed, all of the lines of text in which any string is found or replaced are listed on the screen.

If n, the Find and Replace commands execute as usual, but the lines in which any string is found or replaced are not separately listed on the screen.

- o Tabs - permits review or change of tab settings (default is 4).

----old tabs: #
Tabs:

The message line lists the current tab settings. If you only wish to inspect the tab settings, press CONTROL C to return to main command level.

If you type in numbers increasing in the range 0

to 158, tabs are set as indicated. If the last column is less than 159, the difference between the last two stops is repeated.

For example: 4 sets tabs at 4, 8, 12, 16...
6, 10 sets tabs at 6, 10, 14, 18

Note

Columns start at 0, not 1.

- o Viewrow - specifies the row on the screen in which the cursor should appear after the View command is invoked (default row is 5).

row for view:

Type the number of the line on which you wish the cursor to be positioned by the View command. To center the cursor on the middle lines, set viewrow to 10 or 11. Legal viewrows are row numbers 1-21.

When PSCAN is first invoked, its syntax checking and automatic formatting features are on. To turn both these features off, do the steps in the following example.

Example

Press S. (invokes Set command)
Press E. (invokes Err_Check subcommand)
Type n. (indicates no for syntax checking option)
Press <ESC>. (returns PSCAN to main command level)

The automatic formatting feature is turned off because the syntax checking feature has been turned off. Automatic formatting cannot function without syntax checking.

Find, -find

The Find (-find) commands work at main command level and in the BLOCK and MACRO modes. These command are used to locate a specific string of text (referred to as the target string). A string can be composed of one character or several characters.

Find searches forward for the target string from the cursor position; -find searches backward from the present cursor position for the target string. These commands are useful for finding strings of text quickly.

Both the Find and -find commands accept a count where count indicates the number of times to search for a target string. The search stops after finding the last occurrence of the target string or where count is exhausted.

Example

Press F. (invokes Find command, to invoke -find, press -)

PSCAN prompts -

Find"

Enter the characters of the string you wish to find exactly as they appear in the file. Do not type a <cr> because it will be added to the string as if it were part of the string you wish to find.

Press <ESC>. (PSCAN searches forward from the current cursor position; when it finds the target string it positions the cursor in the first column following the occurrence of the characters of the string.)

Insert

The Insert command enters INSERT mode, in which you can enter text. During INSERT mode, any character keys that are pressed are entered into the edit buffer and displayed on the screen. This command can be entered from main command level or MACRO mode.

Example

Press I. (enters INSERT mode)

The following message appears -

[insert] (to remind you that any characters you type during INSERT mode will be inserted in the text)

Enter the desired text.

This is an example of inserting text.

Press <ESC>.

(returns PSCAN to main command level)

Replace, ?replace

The Replace (?replace) commands are accepted at main command level and in MACRO mode. They are used to substitute a new string for an old string or to delete a string.

The ?replace command functions in the same way as the Replace command, except that it prompts the following for each occurrence of the target string:

```
-----  
ok to replace? (y or [n])
```

The ?replace command ensures that you replace only what you want to replace.

The Replace and ?replace commands accept any count where count indicates the number of times you are prompted to verify replacement. The replacement stops after the last occurrence of the target string.

Examples

1. Press R.

PSCAN prompts -

```
-----  
Replace ""
```

Type in the characters of the target string.

Press <ESC>.

PSCAN prompts -

```
-----  
Replace "string" with ""
```

Type in the string with which you want to replace the old one.

Press <ESC>.

To delete a string using Replace or ?replace, take the steps shown in Example 2.

2. Press ?. (invokes ?replace command)

PSCAN prompts -

?Replace ""

Type in the string of characters you wish to delete.

PSCAN prompts -

?Replace "string" with ""

Do not type anything.

Press <ESC>.

PSCAN prompts -

ok to replace? (y or [n])

Enter y.

The old string is deleted.

Jump

The Jump command works at main command level and in the BLOCK and MACRO modes. It allows you to get to the start or end of the file quickly, to position the cursor in or near a line in which a syntax error has been detected, to position the cursor on a specified line of your program or text, to position the cursor in a specified column on the screen, or to a point in the program that has been "tagged" using the Tag command.

Each of the subcommands that perform the above functions is described below, with any prompts that appear when the subcommand is invoked.

- o Start - moves the cursor to the start of the file.

- o End - moves the cursor to the end of the file.
- o eRror - moves the cursor to a point near the syntax error. (To invoke, press R.)
- o Line - moves the cursor to the designated line.
PSCAN prompts -
line:
- o Position - moves the cursor to the designated column.
PSCAN prompts -
column:
- o A tag B tag C tag D tag - moves the cursor to the tag you set previously using the Tag command (If you did not set any tag previously with the Tag command, this subcommand cannot function.) The Tag command is described later in this chapter.

Example

To move the cursor to line 50 of your file, do the following:

Press J. (invokes the Jump command)

Press L. (invokes the Line subcommand)

PSCAN prompts -

line:

Enter 50 <cr>. (cursor appears on line 50)

BLOCK MODE COMMANDS

The BLOCK mode allows for manipulation and deletion of blocks of text. Text that is delimited or deleted in the BLOCK mode is temporarily stored in the BLOCK buffer or in an external file that you specify. The commands in this mode are especially useful for shuffling lines of code during program reorganization.

To delimit a section of text for transfer to the BLOCK buffer, move the cursor to the first character in the block you wish to manipulate. Press B; @ replaces the character, marking one endpoint of the block.

The menu prompts:

Buffer Delete Find -find Jump Put

Move the cursor to the character immediately after the block being delimited. The character under the first @ sign is included in the block; the character under the second one is not.

Next choose one of the commands described below for specifying exactly what is to be done with the delimited block of text. (The Find, -find, and Jump commands function the same way in BLOCK mode as they do at main command level.)

Buffer

The Buffer command causes the text delimited between the @ signs to be copied to the BLOCK buffer.

Delete

The Delete command removes the delimited section of text from the file and places it in the BLOCK buffer. This text cannot exceed 2 Kbytes. If it exceeds 2 Kbytes, the following message is displayed:

----cannot save in memory save anyway? ([y] or n)

If you press y, only 2 Kbytes of the text are stored in the BLOCK buffer. The remaining bytes are stored on disk. However, all the text is retrievable by use of the Get command.

Put

The Put command copies the text that has been delimited to an external file (which you specify by name). The delimited text is not, however deleted from the current file; a copy of the delimited text is made and is put in the specified file.

Example

Position the cursor over the first character of the block of text you wish to delimit.

Press B.

(enters the BLOCK mode; delimits the first character of the block of text)

Move the cursor to the space just after the last character of the block of text you wish to delimit.

Press P.

PSCAN prompts -

Output file:

Enter the name of the file to which you want the delimited block of text to be copied and then enter a carriage return.

Output file: xxxx <cr>

The block of text is copied to the specified file.

Delete

The Delete command contains subcommands that are the same as the commands available in the BLOCK mode.

This Delete command exists so that you can delete blocks of text by typing D at both endpoints of the block, rather than going into BLOCK mode, delimiting the text and then using the Delete command within the BLOCK mode.

Example

Press D. (invokes Delete command)

PSCAN prompts -

Buffer Delete Find -find Jump Put

Move the cursor to the space just after the last character of the text you wish to delete.

Press D. (the delimited text is deleted)

Get

The Get command functions at the main command level. It retrieves the contents of the BLOCK buffer (or any external file). The Get command places the retrieved text at the current cursor position in your file.

This command accepts any finite count. The content of the BLOCK buffer or the external file is copied to the current cursor position count times. The repeat count / is not valid with the Get command. If / is typed, PSCAN returns to main command level.

Example

Press G. (invokes the Get command)

PSCAN prompts -

input file:

Enter the name of the external file in which you had text stored. If it was stored in the BLOCK buffer, you do not need to enter anything in response to this prompt.

Press <cr>. (PSCAN retrieves the contents of the BLOCK buffer or the named external file and places it at the current cursor position.)

Other

The Other command allows you to switch between PSCAN's main edit buffer and a secondary buffer called the Other buffer. This enables you to examine, search, or copy text from a file stored in the Other buffer while still working on a file stored in the main edit buffer. When Other is invoked, the file in the secondary buffer can be viewed on the screen. To get back to the main edit buffer, press the letter O a second time. In this way you can toggle back and forth between the Other buffer and the main edit buffer. Note that no PSCAN functions are available in the Other buffer.

Example

Press O. (invokes the Other buffer)

Use the Other buffer to edit the

primary buffer by examining searching
or copying from the Other buffer.

Press O again. (returns PSCAN to the primary edit
buffer)

Note that the secondary buffer has a fixed maximum size of 7k.
If text size is greater than 7k and text is lost, the message

----some text lost

appears and the output file is changed to the byte bucket (:BB:) to avoid inadvertent loss of the file with a QUIT Exit or a QUIT Update command.

Tag

The Tag command enables you to identify or "tag" specific locations within a file. Using the Tag command and the Jump command, you can quickly move the cursor to a "tagged" location. Four different locations can be tagged with this command.

Example

Position the cursor at the
location you wish to tag.

Press T. (invokes the Tag command)

PSCAN prompts -

A tag B tag C tag D tag

Press one of the above
letters to identify the
tagged location.

(the letter you press will refer to
the location you have identified)

Undo

This command restores up to 100 characters that were deleted by the last CONTROL Z (Delete Line), CONTROL X (Delete Left), or CONTROL A (Delete Right) command. To invoke this command, press CONTROL U. Consecutive CONTROL Us will repeat the last restoration.

The Undo command does not restore text removed by the use of the Delete command in BLOCK mode. These characters are

stored in the BLOCK buffer and can be retrieved using the Get command.

Example

Press CONTROL U. (invokes the Undo command)

PSCAN restores up to 100 characters that were last deleted.

View

This command is invoked from the main command level and MACRO modes. It rewrites the screen and repositions the cursor and the line of text in which it is currently located to the viewrow line. The default viewrow is 5; this can be changed using the Set command (described earlier in this chapter).

Example

Press V. (invokes the View command)

PSCAN repositions the cursor and the line in which it is positioned to the viewrow line.

Again

The Again command simply repeats the action of the previous command except for mode-switching commands like INSERT or XCHANGE. It is invoked from the main command level. If Again is invoked after a Find, -find, Replace or ?replace, the arguments of the previous command are also repeated. Count for the Again command is used as count for the repeated command; the count of the last command is ignored.

Hex

This command inserts the ASCII equivalents of hexadecimal values in the text. It can also be used to display the hexadecimal values of the character(s) indicated by the cursor on the message line of the screen.

Any [count] that precedes the Hex command gives the number of bytes to be displayed in hexadecimal format. Up to 10 bytes of hexadecimal values can be displayed in the message line. If more bytes are to be displayed, the message "hit space to continue" is displayed. Press the space bar when this message appears and the next 10 bytes are displayed; any other key returns PSCAN to the main command level.

Examples

1. Position the cursor at the location you wish the hex value to be inserted.

Press H. (invokes the Hex command)

PSCAN prompts -

Input Output

Press I. (indicates to the PSCAN that you want it to insert the ASCII equivalent of a hexadecimal value that you enter)

PSCAN prompts -

Input:

Enter the hexadecimal value that you want inserted at the current cursor position.

0C (PSCAN inserts the hex value 0C, or form feed, at the current cursor position. A question mark (?) appears on the screen at the current cursor position because form feed (0C) has no printed representation.)

2. Position the cursor over the character for which you want the hex value displayed.

Press H. (invokes the Hex command)

PSCAN prompts -

Input Output

Press O. (indicates to the PSCAN that you want it to display the hexadecimal value of the character over which the cursor is currently positioned)

PSCAN displays the hexadecimal value of the specified character(s) on the message line of the screen:

-----xx

Xchange

The Xchange command causes the editor to enter the XCHANGE mode. This is the mode in which you can type in new characters over old ones. XCHANGE mode can be entered from main command level and MACRO modes.

Example

Press X. (enters XCHANGE mode)

The following message appears -

[exchange] (to remind you that any characters you type will replace the characters over which the cursor is positioned)

Move cursor to be just over the beginning of the text displayed on the screen that you wish to change.

Type in desired text.

Press <ESC>. (to return to main command level)

PL/M-86 ORIENTED COMMANDS

The following six commands are very important to PSCAN because they give it power as a PL/M-86 syntax checking editor. Each of these commands is invoked by pressing the CONTROL key and a specific alphanumeric key. The control characters that perform these commands are configurable. The following explanations list

the control characters that perform these commands on a Series III terminal. Check the macro file for your specific terminal to determine which control characters perform these commands on your terminal.

Move-Level-Out

The Move-Level-Out command moves the cursor to the beginning of the enclosed block in which the cursor is currently positioned. To invoke, press CONTROL T.

Move-Level-In

The Move-Level-In command moves the cursor to the beginning of the first enclosed block. To invoke, press CONTROL Y.

Example

Press CONTROL Y. (invokes Move-Level-In command)

Figure 3-1 shows that when the Move-Level-In command is invoked, the cursor moves to the beginning of the first enclosed block.

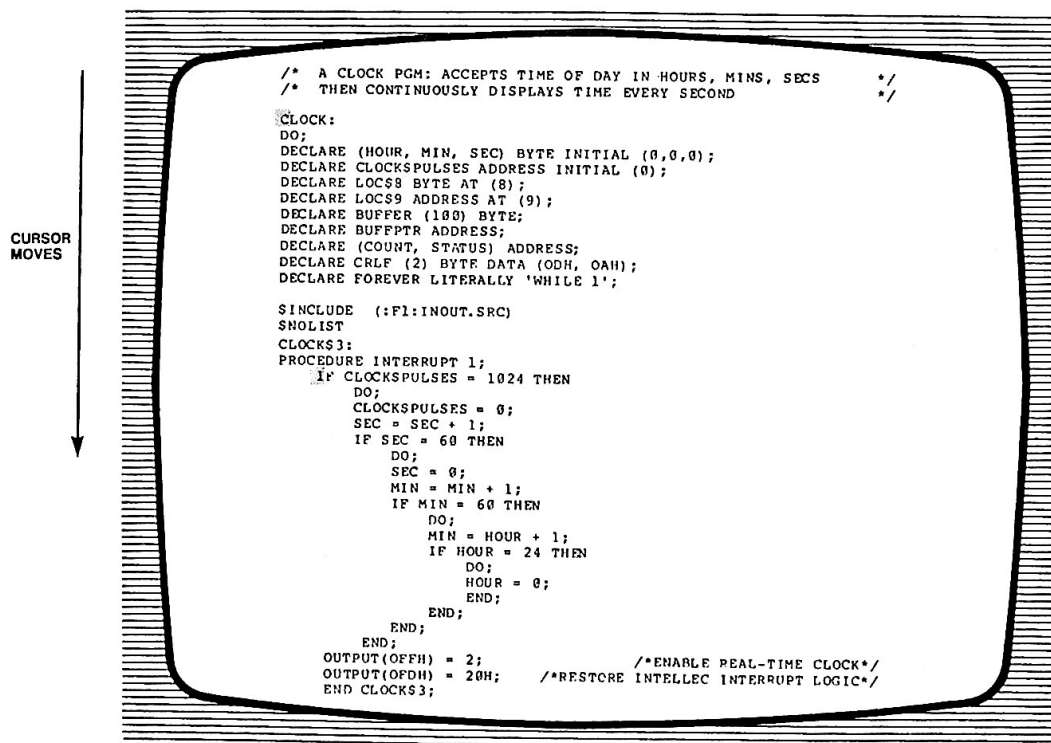


Figure 3-1. Move-Level-In Command

Move-Statement-Up

The Move-Statement-Up command moves the cursor back one statement (at the same indentation level from which it is moved). To invoke, press CONTROL B.

Move-Statement-Down

This command moves the cursor forward one statement (at the same indentation level from which it is moved). To invoke, press CONTROL N.

Example

Press CONTROL N. (invokes Move-Statement-Down command)

As shown in Figure 3-2, the cursor moves forward one statement.

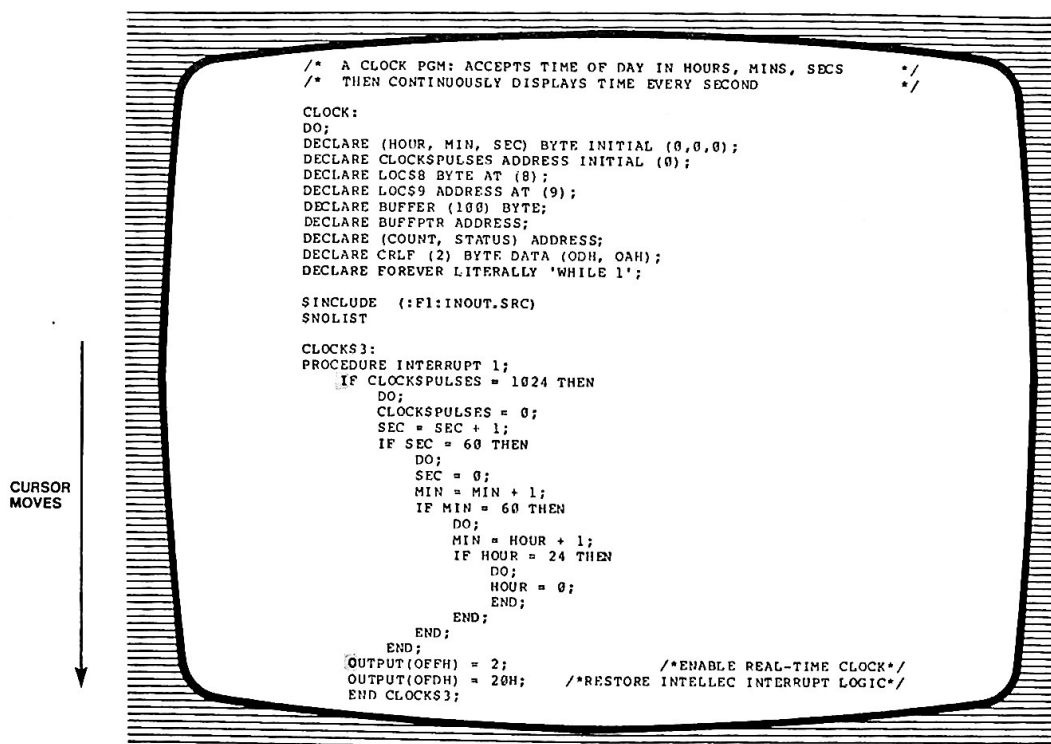


Figure 3-2. Move-Statement-Down Command

Move-Token-Left

The Move-Token-Left command moves the cursor back one token (or smallest meaningful unit of a statement). To invoke, press CONTROL O.

Move-Token-Right

The Move-Token-Right command moves the cursor forward one token. To invoke, press CONTROL P

Example

Press CONTROL O. (Invokes the Move-Token-Left command)

As shown in Figure 3-3, the Move-Token-Left command moves the cursor one token to the left.

```

/* A CLOCK PGM: ACCEPTS TIME OF DAY IN HOURS, MINS, SECS
/* THEN CONTINUOUSLY DISPLAYS TIME EVERY SECOND */

CLOCK:
DO;
DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0,0,0);
DECLARE CLOCKSPULSES ADDRESS INITIAL (0);
DECLARE LOC88 BYTE AT (8);
DECLARE LOC29 ADDRESS AT (9);
DECLARE BUFFER (100) BYTE;
DECLARE BUFFPTR ADDRESS;
DECLARE (COUNT, STATUS) ADDRESS;
DECLARE CRLF (2) BYTE DATA (0DH, 0AH);
DECLARE FOREVER LITERALLY 'WHILE 1';

$INCLUDE (:F1:INOUT.SRC)
$NOLIST

CLOCK$3:
PROCEDURE INTERRUPT 1;
IF CLOCKSPULSES = 1024 THEN
DO;
CLOCKSPULSES = 0;
SEC = SEC + 1;
IF SEC = 60 THEN
DO;
SEC = 0;
MIN = MIN + 1;
IF MIN = 60 THEN
DO;
MIN = HOUR + 1;
IF HOUR = 24 THEN
DO;
HOUR = 0;
END;
END;
END;
END;
END;
OUTPUT(OFPH) = 2; /*ENABLE REAL-TIME CLOCK*/
OUTPUT(OPDH) = 20H; /*RESTORE INTELLEC INTERRUPT LOGIC*/
END CLOCK$3;

```

Figure 3-3. Move-Token-Left Command

QUIT MODE COMMANDS

The commands used in QUIT mode end an editing session in one of several ways. These commands are described below, with any prompts that appear when they are invoked.

The following menu prompt line appears after QUIT mode is entered -

```
----  
Abort  Exit  Init  Update  Write
```

Abort

The Abort command returns PSCAN to the operating system level without saving changes made to the text in the edit buffer during the session. It displays the following prompt in order to ensure that you do want to return to system level without saving changes. If you enter "n" instead of "y", PSCAN returns to main command level.

```
----  
all changes lost? (y or [n])
```

Exit

The Exit command saves the content of the edit buffer in the output file. If the input and output files are the same, the content of the input file is overwritten. PSCAN returns to operating system level and a backup file called FILENAME.BAK is also created with this command, unless the BAK option has been turned off with the Set command.

Init

This command allows you to start a new editing session without having to return to the system level first. Before initializing the new editing session, if any changes have been made to the file and the file has not been saved with the Update command, the following prompt appears:

```
----  
all changes lost? (y or [n])
```

Entering n ensures that any changes made during the latest editing session are saved before a new editing session is started and it returns PSCAN to main command level.

If you enter y to the Init prompt, the following prompt appears next -

enter [file[To file]]

This indicates that you need to enter the name of the new file you wish to edit and the (optional) filename to which you want the new file written when you exit.

Update

This command updates the file you are working on without returning you to the system level. The following message appears after the Update command is invoked:

----- filename has been written

Press <ESC> to return to the main command level, from which the editing session can be continued.

Write

This command allows you to write the entire content of a file to a specific file (which you name), without returning to operating system level.

The following prompt appears for you to enter the name of the file to which you want the present file content written.

Output file:

If you have been editing a new file and have not specified a filename, the following message and command choices appear:

-----no input file

Abort Init Write

Press one of the command choices or <ESC> or CONTROL C to return to main command level.

Whenever QUIT Abort or QUIT Exit is invoked, PSCAN checks the Other buffer for any changes. If changes are found, the Other buffer displays the following message and prompt:

-----Other editing filename [to filename]
all changes lost? (y or [n])

If you do not wish to save the changes, type y. The following message is displayed:

----Filename has been written

An n response returns PSCAN to main command level in the Other edit buffer. At this point, press Q for QUIT mode and U for the Update command. PSCAN returns to main command level and the following message is displayed:

----Filename has been written

Example

Press Q. (enters the QUIT Mode)

PSCAN prompts -

Abort Exit Init Update Write

Press E. (invokes the Exit command)

The following message appears -

Filename has been written

See Chapter 4 for information on the MACRO mode and commands and the Execute command.

COMMANDS AVAILABLE WITHIN MODES

Table 3-1 lists all of the PSCAN commands (except for MACRO mode commands) in alphabetical order in the left-hand column and the particular modes in which the command is accepted in the right-hand column.

Table 3-1. Commands Available within Modes

PSCAN Command	Mode(s) Available In
Abort	QUIT
Again	MAIN COMMAND LEVEL, BLOCK,
Buffer	BLOCK
Delete	MAIN COMMAND LEVEL, BLOCK
Exit	QUIT
Find, -find	MAIN COMMAND LEVEL, BLOCK,
Get	MAIN COMMAND LEVEL
Hex	MAIN COMMAND LEVEL
Init	QUIT
Insert	MAIN COMMAND LEVEL, (enters INSERT mode)
Jump	MAIN COMMAND LEVEL, BLOCK
Other	MAIN COMMAND LEVEL

Table 3-1. Commands Available within Modes (Cont'd.)

PL/M-86 oriented commands	MAIN COMMAND LEVEL, INSERT, XCHANGE, BLOCK
Put	BLOCK
Replace, ?replace	MAIN COMMAND LEVEL
Set	MAIN COMMAND LEVEL
Tag	MAIN COMMAND LEVEL
Write	QUIT
Undo	MAIN COMMAND LEVEL
Update	QUIT
View	MAIN COMMAND LEVEL
Write	QUIT
Xchange	MAIN COMAND LEVEL, (enters XCHANGE mode)

DIRECTIONAL ARROW KEYS AND DELETE SPECIAL-PURPOSE KEYS

The directional arrow keys are accepted in all modes.

Special-purpose delete keys are accepted at main command level and in the INSERT and XCHANGE modes.

A count preceding a directional arrow key is accepted only at the main command level and in BLOCK and Macro modes. A count preceding a special-purpose delete key is accepted at main command level only.



•

•



•

•



INTRODUCTION

In addition to the modes and commands described in Chapter 3, PSCAN has a MACRO mode that accepts five commands for creating and manipulating macros. These commands, in addition to the Execute command, are described in detail later in this chapter.

A macro is a sequence of commands, written in a special set of symbols. Macros are generally created to avoid repeatedly typing in a long sequence of commands. PSCAN's macro facility speeds your work and reduces the typing errors associated with entering long command sequences. Instead of entering the desired command sequence, you can simply invoke a macro (which you have already created and named, and which is either stored in the macro storage or in an external file), and the commands that the macro represents are executed.

MACRO mode is entered like the other PSCAN modes, by pressing the first character of the mode from main command level. On a Series IV, MACRO mode can also be invoked by pressing the function key on the keyboard that appears under the word "Macro" displayed in the menu prompt line.

When you enter MACRO mode, the following menu prompt line appears on the display screen:

```
----  
Create  Get  List  Insert  Save
```

These macro commands are invoked similarly to the other PSCAN commands, by pressing the first character of the command (or by pressing a function key on the Series IV).

The following section provides a general description of how macros are created, stored and retrieved. Examples of macros and macro creation are also provided in this chapter.

MACRO CREATION

Creating a macro involves two steps. First, the macro must be both defined, using the special set of symbols that represent macros, and named. Next, the macro must be stored in some way, either in the macro storage area of the edit buffer, or for permanent storage, in an external file that you name.

MACRO NAMES

You must name each macro you create so that later you can execute it.

Macro names can be character strings or single characters, including control characters. The way in which a macro is named determines how it can be executed:

- o Any macro, no matter how it is named, can be executed from main command level, using the Execute command.
- o Macros named with single characters that are not the initial letters of PSCAN commands can also be executed when the macro name is typed in at main command level.
- o Macros named with single control characters (e.g., CONTROL P, CONTROL Y) that are not PSCAN commands can be executed by entering its name at main command level or when PSCAN is in the INSERT or XCHANGE mode.

If a macro name is used to create a macro more than once during an editing session, PSCAN will execute only the macro most recently placed in macro storage. When a name is reused, PSCAN substitutes the new command sequence for the old command sequence, effectively deleting the old macro. Unless the macro has already been written in macro form in a macro file, or into a text buffer with Macro Save, the first macro definition will be lost.

MACRO FILES

A macro file can contain macros, Set commands, configuration commands, and user comments.

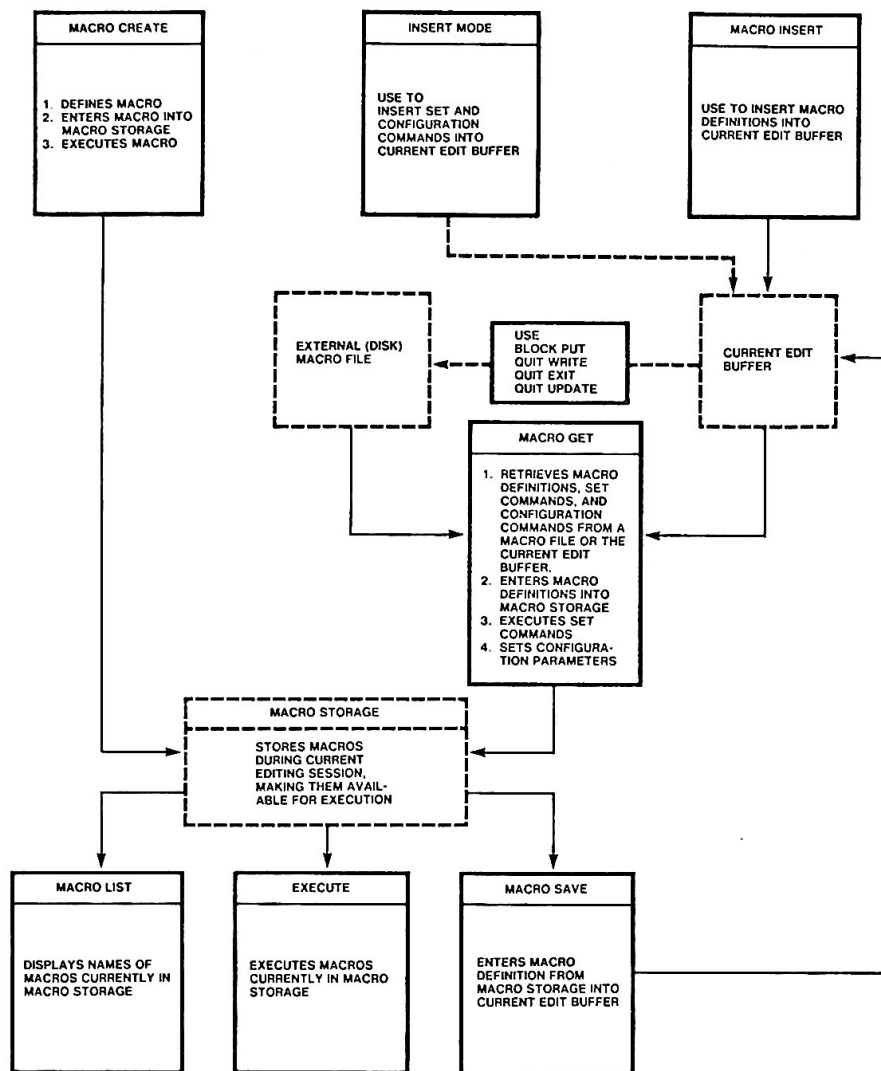
When PSCAN is invoked, it tries to read a macro file named PSCAN.MAC, unless you specify a different macro file or you specify the NOMR control in the invocation line. PSCAN.MAC contains configuration commands for your particular computer terminal. Macro definitions can be stored in the PSCAN.MAC file or in additional macro files that you create. These macros are later read into macro storage.

When you want to execute a macro that is in a macro file, you must first retrieve the macro definition from the macro file and put it into macro storage using the Get command. Then, using the Execute command (at main command level) you can execute any macro in the given macro file.

Comments (describing a macro's function, for example) can be inserted in macro files before or after macro definitions. Comments can be longer than one line. They must begin with the characters `*` and end with the characters `*\`.

You can use up to 1 Kbytes of memory for definitions in macro storage. If the macro definition characters in macro storage exceed this limit, the message "no more room for macros" appears in the message line when you use the Create or Get commands. To make more room available in macro storage, you can delete macros that are no longer needed.

Figure 4-1 represents the various commands and files involved in macro creation and use. The arrows indicate a flow of activity that occurs when the particular command is invoked. Boxes drawn with dotted lines indicate the dynamic nature of the particular enclosed area (for example, the macro storage area changes continually, depending on what is stored there at a particular time). Dotted arrows indicate that the flow of activity is optional.



NOTES:

- CREATE MACROS WITH MACRO CREATE OR MACRO INSERT.
- USE MACRO INSERT TO ENTER MACRO DEFINITIONS IN MACRO FORM.
- USE MACRO CREATE TO TYPE COMMANDS IN A MACRO AS AT MAIN COMMAND LEVEL.
- USE INSERT MODE TO ENTER SET COMMANDS, CONFIGURATION COMMANDS, AND COMMENTS.
- MACROS MUST BE IN MACRO STORAGE TO BE AVAILABLE FOR EXECUTION.
- TO PERMANENTLY SAVE MACROS CREATED WITH MACRO CREATE, WRITE THE MACROS TO THE CURRENT EDIT BUFFER WITH MACRO SAVE. THEN USE THE BLOCK PUT, QUIT WRITE, QUIT UPDATE, OR QUIT EXIT COMMANDS TO WRITE THE CONTENTS OF THE CURRENT TEXT BUFFER TO AN EXTERNAL FILE.

Figure 4-1. Macro Commands and Processes

Macro Examples

This section presents a few examples of macro creation. Usually, interactive macro creation is the preferred method of creating macros, it is shown in the following examples before the commands are described in detail.

Example 1

When writing in a block-structured language like PL/M-86, it is often necessary to move a block of code to the right or left to adjust the indentation. The following macros will adjust text where tabs are used for indentation.

To define these macros interactively using the Create command, enter the following commands:

Press M. (to enter MACRO mode)

Press C. (to invoke the Create command)

PSCAN prompts -

Macro name:

Enter the macro name -

> <cr> (names macro >)

Press ↑. (to move the cursor up one line)

Press R. (to invoke the Replace command)

PSCAN prompts -

----Macro

Replace "" with ""

Press <cr> <TAB> <ESC>. (fills in prompt with commands to be replaced, and displays remainder of prompt)

<cr> <TAB> <TAB> <ESC> (completes prompt line with replacement commands)

Press ↓. (moves cursor down one line)

Press M. (terminates macro definition)

You can now use the MACRO Save command to save the macro file.

To do this, retrieve the macro from macro storage into the edit buffer (the Other buffer in this example). Write the contents of this buffer into the file in which you want to save the macro (a file named Junk in this example).

Press O. (to enter the Other buffer)

Press Q. (to enter QUIT mode)

Press I. (to invoke the Init command)

PSCAN prompts -

enter [file[To file]]

Enter filename:

Junk

Press M. (to enter MACRO mode)

Press S. (to invoke the Save command)

PSCAN prompts -

Macro name:

Enter the name of the
macro you wish to save -

Press > <cr>. (> is the name of the macro to be saved)

Press Q. (to enter QUIT mode)

Press U. (to invoke the Update command; ">" is now
saved in the file named Junk)

The macro > can now be used in future editing sessions. To make macro > available for execution during a future editing session, you would use the Get command. In response to the prompt "macro file:", you would enter the filename in which you saved the macro.

To create these macros using the Insert command, invoke Insert from MACRO mode, then type the following keys without including blanks:

M > ESC ↑ R \ N L TAB ESC \ N L TAB TAB ESC ↓ \ E M
 <cr>

M < ESC ↓ R \ N L TAB ESC \ N L ESC ↑ \ E M <cr>

As you type, PSCAN inserts the following in the current edit buffer:

M>\BR\CUR\NL \BR\NL \BR\CD\EM
 M<\BR\CUR\NL \BR\NL\BR\CD\EM

Note that in MACRO mode, during the Insert command, tabs appear as a series of blanks, and that you must type the character sequence \NL to insert a carriage return in a macro definition. You must also type the character sequence \EM.

To make these macros available for execution, type M, G and <cr> or <ESC>.

You can now execute these macros in several ways. You can type E(XECUTE) and a macro name (either > or <). Or, you can type just the one-character macro name while at main command level.

In either case, executing the macro ">" will add a tab to the next line in the current file that contains tabs. Executing the macro "<" will delete a tab from the next line that contains tabs. Typing a repeat count and then the macro name will execute the macro count times. For example, type "10>" to indent the next 10 lines.

Example 2

Suppose that you want the following macros, named "." and ",", to move the cursor eight characters to the right and eight characters to the left, respectively.

To define these macros interactively using the MACRO Create command, type the following:

Press M. (to enter MACRO mode)

Press C. (to invoke the Create command)

Enter , or . <cr>.

Enter 8 → . ← (moves cursor eight times to the right)
 or 8 ← (moves cursor eight times to the left)

Press M. (to terminate macro definition)

After you define these macros, typing a period (.) at main command level will move the cursor eight characters to the right;

typing a comma (,) will move the cursor eight characters to the left.

To define these using MACRO Insert, press the following keys:

```
M . <ESC> 8 → \ E M <cr>    and
M , <ESC> 8 ← \ E M ;
```

The macro definitions appear in your macro file as follows:

```
M.\BR8\CR\EM
M,\BR8\CL\EM;
```

You can add comments to a macro file. For example, using INSERT mode, type the following comment next to the macro definition for the macro ",":

```
\*Moves cursor 8 spaces to left*\
```

Example 3

You can use macros to repeatedly insert character strings in text. For example, the following macro will insert a string of hyphens (---) at the cursor position in the current text.

To create this macro interactively, use the following commands:

```
Press M.                (to enter MACRO mode)
Press C.                (to invoke the Create command)
Enter @ <cr>.           (to name the macro @)
Enter I ----- <cr>.
Press M.                (to terminate macro)
```

After you define this macro, typing "@" at main command level will insert a string of hyphens at the current cursor position.

To enter this directly to a macro file, use MACRO Insert, and press the following keys:

```
M @ <ESC> I ----- \NL <ESC> \EM;
```

Note that a semicolon (;) signals the end of the above macro. Pressing <cr> would have the same effect.

Macro Definitions

As shown in the above examples, macros can be defined in two ways:

- o The Create command defines a macro, puts the definition into macro storage, and executes the macro.
- o The Insert command allows you to type macro definitions directly into the current edit buffer.

All macro definitions are stored in macro form, using the symbols shown in Table 4-1.

Table 4-1. Macro Form Symbols

Symbol	Command
\BR	ESCAPE
\CU	UP
\CD	DOWN
\CR	RIGHT
\CL	LEFT
\CH	HOME
\LI	MOVE-LEVEL-IN
\LO	MOVE-LEVEL-OUT
\SU	MOVE-STATEMENT-UP
\SD	MOVE-STATEMENT-DOWN
\TL	MOVE-TOKEN-LEFT
\TR	MOVE-TOKEN-RIGHT
\XA	DELETE RIGHT (CONTROL A)
\XF	DELETE CHAR (CONTROL F)
\XU	UNDO (CONTROL U)
\XX	DELETE LEFT (CONTROL X)
\XZ	DELETE LINE (CONTROL Z)
\NL	CARRIAGE RETURN
\RB	RUBOUT
\0hh	Control characters, where hh is the character's hexadecimal value
\EM	End of macro definition

Macro definitions must take the following form:

```
M    macro name \BR    characters in macro \EM {<cr> | ;}
```

Where

M declares that a macro definition follows.

macro name is any name given to the macro being defined.

`\BR` stands for the ESC key, and signals the end of the macro name.

characters in macro are a sequence of commands to be executed.

`\EM {<cr>|;}` signals the end of the macro definition. A carriage return (<cr>) or a semicolon (;) must follow `\EM`.

Example

The following macro definition appears as it would in a macro file:

```
MMACRO1\BR/RABC\BRDEF\BR\EM;
```

Where

`M` declares that a macro definition follows.

`MACRO1` is the macro name.

`\BR` must follow a macro name.

`/RABC\BRDEF\BR` are the characters in the macro. They represent the following:

`/` is the forever count, which makes the next command execute throughout the current file from the current cursor position to the end-of-file.

`R` is the Replace command.

`ABC` are the characters that the macro will replace.

`\BR` is the ESC key. During normal PSCAN use, this causes PSCAN to prompt

with "

and to wait for you to type in replacement characters. During macro execution, the same sequence must occur, so <ESC> must be inserted.

`DEF` are the characters that will replace ABC.

`\BR` is the <ESC> that executes the Replace function.

`\EM;` ends the macro definition.

When this macro is invoked, it replaces the characters "ABC" with the characters "DEF" throughout the current file, from the current cursor position to the end-of-file.

Set Commands In Macro Files

To enter Set commands in a macro file, use INSERT mode (not MACRO mode). Type the same sequence of characters that you would type if you were invoking the command at main command level. Include responses to prompts. Terminate the Set command with a carriage return (<cr>) or a semicolon (;).

Example

The following Set commands appear as they would in a macro file:

SAY; stands for Set Autocr yes. This turns on Autocr.

ST 8; stands for Set Tabs 8. This sets tabs at every eighth column.

MACRO COMMANDS

Create

As shown in an earlier examples, the Create command allows you to create a macro interactively during the current editing session. This command performs three things simultaneously: it defines a macro, puts in into macro storage, and executes it.

Example

Press M. (to enter MACRO mode)

Press C. (to invoke the Create command)

PSCAN prompts -

Macro name:

<cr> (Enter the name you want for the macro followed by <cr> or <ESC>.)

The menu prompt line in Figure 4-2 appears on the screen:

```
-----  
Again  Block  Delete  Execute  Find  -find  Get  --more--
```

Figure 4-2. Menu Prompt Line

PSCAN is still in MACRO mode, although the commands displayed on the menu prompt line are the same as those displayed at main command level. This is because these same commands are available to you during macro creation.

Enter the keystrokes necessary to complete the task just as you would at main command level.

(PSCAN "remembers" or "traps" these keystrokes; they constitute the commands that will execute when you use this macro.)

Press M.

(To terminate macro creation and return to main command level. Pressing CONTROL C terminates macro creation without saving the macro. Pressing M terminates macro creation and saves the macro in macro storage.)

Once a macro has been created, you can use the Save command (described later in this chapter) to write it to the current edit buffer. You can then save the macro by using a BLOCK Put command or a QUIT mode command. Otherwise the macro just created will be lost when you exit PSCAN.

Get

The Get command retrieves macro definitions, Set commands and configuration commands from an external file or the current edit buffer. It also enters the retrieved macro definitions into macro storage, executes the retrieved Set commands, and sets the configuration parameters specified by the retrieved configuration commands.

The Get command makes previously defined macros available for execution during the current editing session.

Example

Press G. (to invoke the Get command)

PSCAN prompts -

Macro file:

Mac.1 <cr> (Enter the filename in which the macro
you wish to retrieve is stored, followed
by a <cr>. PSCAN reads the Mac.1 file
as a macro file.)

If you do not enter a filename, but simply press <cr> or <ESC>, PSCAN reads the current text as a macro file.

By default, PSCAN will process the macro file named PSCAN.MAC upon invocation. Therefore you do not have to use the Get command to execute the macros within PSCAN.MAC. Before you can execute the macros stored in other macro files, you must retrieve the files with the Get command, unless you have specified the macro filename when you invoked PSCAN.

Insert

As shown earlier in this chapter, the Insert command enables you to create macros by inserting commands, in macro form, directly into the current edit buffer. This command causes all subsequent input (including <RUBOUT>) to be inserted in the text in macro form.

Typing CONTROL C terminates the Insert command.

Press M. (to enter MACRO mode)
Press I. (to invoke the Insert command)
PSCAN prompts -

Control C to stop

Type in the macro definition (do not enter spaces between characters): (When you press <ESC> and <RUBOUT>, Control characters and directional arrow keys, the macro form symbols for these keys appear on the screen.)

MCONTROL<ESC>→←<HOME>\EM<cr> (As you type, PSCAN adds the following macro definition at the current cursor location:

M\OOC\BR\CR\CL\CH\EM

This macro defines CONTROL L to mean jump to start of line.)

Press CONTROL C. (to terminate the Insert command)

Pressing <cr> moves the cursor to a new line in the current text. To make a carriage return/linefeed insertion part of a macro, you must type the characters \NL (new line) individually. You must also type the characters \EM (end macro) individually. If, as part of the macro you want PSCAN to insert a backslash (\), you must type the backslash twice.

List

The List command displays the names of all the macros that are currently in macro storage and available for execution. These macro names appear in the message line. If the message line is not large enough to accommodate the list of all of the available macros, the message "hit space" appears on the prompt line. To continue the list, press the space bar. Any other command returns PSCAN to main command level.

Example

Press M. (to enter MACRO mode)

Press L. (to invoke the List command)

PSCAN displays a list of macro names that are currently in macro storage and available for execution.

Save

The Save command translates macros defined interactively with the Create command into macro form and writes the macro definition into the current edit buffer.

Example

Press M. (to enter MACRO mode)

Press S. (to invoke the Save command)

PSCAN prompts -

Macro name:

Type in the macro name followed by <ESC> or <cr>. (If the macro exists, it is placed in the text at the current cursor location in macro form. If it does not exist, the message "no such macro" appears and PSCAN returns to main command level.)

To save a macro created with the Create command, use the QUIT Init command to make a macro file in the current file (by specifying the macro filename when prompted). Then, translate the macro into macro form using the Save command (Type M, then S) and resave the macro file using the QUIT Update or QUIT Exit command.

You can also save the macro to the current text buffer and then write the macro to another file using the BLOCK Put or QUIT Write commands.

Example

The following command sequence saves a macro named * that was already defined with the Create command.

Press Q. (to enter QUIT mode)

Press I. (to invoke the Init command)

PSCAN prompts -

filename:

PSCAN.MAC <cr> (specifies PSCAN.MAC as the input file to edit)

Press M. (to enter MACRO mode)

Press S. (to invoke the Save command)

Press *. (names macro to be saved)

Press <cr>. (executes the Save command, which then translates the macro named * to macro form)

Press Q. (to enter QUIT mode)

Press U. (to invoke the Update command, which updates the macro file PSCAN.MAC to include the macro named *)

Execute Command

The Execute command must be invoked from main command level. It is used to call a macro by name and then execute it. During macro execution, commands execute just as they would if you entered them directly from the keyboard in the same sequence.

Some prompts may prompt you for a response (?Replace, QUIT mode commands, BLOCK Delete, and BLOCK Put) or with the message "hit space to continue." The prompt will appear in the message line, and the macro will pause until you have responded. Then the macro will continue to execute.

Example

Press E. (from main command level) (to invoke the Execute command)

PSCAN prompts -

Macro name:

Enter the name of	(If the macro exists, it is executed.
the macro followed	If it does not exist, the message "no
by <cr> or <ESC>.	such macro" appears on the message
	line.)

The macro terminates execution when it has been fully executed, when an attempt is made to move the cursor (forward, right, down, <HOME>, or <cr>) at the end of the file, when an attempt is made to move the cursor backwards at the beginning of the file, or when a Find, -find, Replace, or ?replace command fails to find its target.

These terminations return PSCAN to main command level unless the macro is nested (i.e., contained in another macro). In this case, PSCAN returns to the next outer layer of the nested macros. Typing CONTROL C during macro execution terminates all nested macros and returns PSCAN to main command level.

Using Macros in INSERT and XCHANGE Modes

PSCAN will execute macros with Control character names while it is in the INSERT or XCHANGE modes. However, if the macro contains main command level commands, then an <ESC> must precede the menu prompt command in the macro. This returns PSCAN from the current mode (INSERT or XCHANGE) to main command level, so that PSCAN reads the character signifying the menu prompt command as a command. For the editor to return to the original mode (INSERT or XCHANGE) after the macro executes, the macro

must also include the INSERT or XCHANGE command. For example, the following macro named CONTROL Q (shown here in macro form) will execute while PSCAN is in the INSERT mode:

```
M\011\BR\BRRJANUARY\BRFEBRUARY\BRI\EM
```

Where

\011	is a backslash and the hexadecimal value of CONTROL Q.
\BR (first)	ends the macro name and thus begins the macro definition.
\BR (second)	returns the PSCAN from INSERT mode to main command level.
R	stands for the REPLACE command.
JANUARY	is the target string searched for by the REPLACE command.
\BR (third)	the third \BR divides the replaced string from its replacement;
FEBRUARY	replaces JANUARY in the current text file.
\BR	the fourth \BR terminates the REPLACE command; returns PSCAN to INSERT mode.
I	
\EM	ends the macro.

NOTE

If a macro with a Control character name is written for use from main command level, and is executed while the editor is in INSERT or XCHANGE mode, the macro will insert (or exchange) the entire string of characters in the macro (including command letters) until it reaches an ESC (\BR).

DELETING MACROS

To delete a macro from macro storage, use the Create command. Type the following:

Press M. (to enter MACRO mode)

Press C. (to invoke the Create command)

Enter macro name <ESC>.

Press CONTROL C.

This will not delete the macro from the macro file, however.

To delete a macro from the macro file, use one of the special-purpose delete keys (for example <RUBOUT>) or the appropriate

control characters (for example CONTROL A) or function keys on your system that perform delete functions.



.

8



.

.



INTRODUCTION

The language oriented editing features of the PL/M-86 syntax checking editor have been mentioned throughout this manual, but they require more detailed attention. This chapter focuses on the language oriented editing features that make PSCAN unique.

The purpose of PSCAN is to provide editing support for the construction and manipulation of PL/M-86 source files. By using PSCAN, you should be able to improve both the quality of a PL/M-86 program and the efficiency of its development.

SYNTAX CHECKING

PSCAN's syntax checking feature, which can be turned on or off with the Set command (described in Chapter 3) helps to ensure that your program is developed free of syntax errors. PSCAN notifies you when and where it detects syntax errors in your program by displaying a "syntax error" message. It is then up to you to determine exactly what the syntax error is and to correct it. PSCAN considers language expressions that do not make sense to be syntax errors. For example, if an END statement is entered without a semicolon following it, PSCAN displays the syntax error message. The rules governing the PL/M-86 language require that a semicolon follow an END; statement.

When the syntax checking feature is turned on (as is its default condition) and PSCAN detects an error, it stops checking the program for further syntax errors until the error already detected has been corrected. This does not mean that you are required to correct a syntax error as soon as it is found; it simply means that the rest of the program being entered will not be checked for syntax until the existing syntax error is corrected. When the error is corrected, PSCAN goes back to the beginning of the program and starts again to check for syntax errors.

For example, during an editing session, when a syntax error is detected by the syntax checking feature, a screen display like the one shown in Figure 5-1 would appear.

```

CLOCK:
DO;
DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0, 0, 0)
DECLARE CLOCK$PULSES ADDRESS INITIAL (0);
DECLARE LOC$8 BYTE AT (8);
DECLARE LOC$9 ADDRESS AT (9);
DECLARE BUFFER(100) BYTE;
DECLARE BUFFER ADDRESS;
DECLARE (COUNT, STATUS) ADDRESS;
DECLARE CRLF (2) BYTE DATA(0DH, 0AH);
DECLARE FOREVER LITERALLY 'WHILE 1';

$INCLUDE(:F1:INOUT.SRC)
$NOLIST

CLOCK$3:
PROCEDURE INTERRUPT 1;
IF CLOCK$PULSES = CLOCK$PULSES + 1;
IF CLOCK$PULSES = 1024 THEN
DO;
CLOCK$PULSES = 0
SEC = SEC + 1;
IF SEC = 60 THEN

----syntax error

```

Figure 5-1. Syntax Error Checking

To determine where the syntax error is in the screen in Figure 5-1, you would invoke the Jump command (by pressing J) and then the eRror subcommand (by pressing R).

The cursor then "jumps" either to the line containing the detected syntax error or to a line near the line containing the syntax error. (See Chapter 3 for a detailed description of the Jump command).

In Figure 5-1, the cursor jumps to a line near the line ending with (0, 0, 0). This line should end in a semicolon for proper syntax.

To correct the error, enter the INSERT mode (by pressing I) and enter a ; at the end of the third line.

The syntax error disappears and PSCAN starts to check the program again (from the beginning of the file) for syntax errors.

Because the syntax error has been corrected, the automatic formatting feature is also enabled and PSCAN starts to format the program.

AUTOMATIC FORMATTING

If your program has been checked for syntax errors and it contains errors that have not been fixed, PSCAN will not format it with its automatic formatting feature.

A program that does not contain syntax errors is automatically formatted with the formatting feature included in PSCAN. Like syntax checking, the formatting feature can be turned on or off with the Set command (the automatic formatting feature's default condition is also "on").

The automatic formatting feature ensures that your program is formatted consistently, and that a program containing syntax errors is not formatted. This feature frees you from having to think about indentation as you are entering your program.

Figure 5-2 is an example of the CLOCK program, formatted with the automatic formatting feature.

```

CLOCK:
DO;
DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0, 0, 0);
DECLARE CLOCK$PULSES ADDRESS INITIAL(0);
DECLARE LOC$8 BYTE AT (8);
DECLARE LOC$9 ADDRESS AT (9);
DECLARE BUFFER (100) BYTE;
DECLARE BUFFPTR ADDRESS;
DECLARE (COUNT, STATUS) ADDRESS;
DECLARE CRLF (2) BYTE DATA(ODH, OAH);
DECLARE FOREVER LITERALLY 'WHILE 1';

$INCLUDE(:F1:INOUT.SRC)
$NOLIST

CLOCK$3:
PROCEDURE INTERRUPT 1;
  CLOCK$PULSES = CLOCK$PULSES + 1;
  IF CLOCK$PULSES = 1024 THEN
    DO;
      CLOCK$PULSES = 0;
      SEC = SEC + 1;
      IF SEC = 60 THEN
        DO;
          MIN = MIN + 1;
          IF MIN = 60 THEN
            DO;
              MIN = 0;
              HOUR = HOUR + 1;
              IF HOUR = 24 THEN
                DO;
                  HOUR = 0;
                END;
              END;
            END;
          END;
        END;
      END;
    END;
  END;
  OUTPUT(OFFH) = 2; /*ENABLE REAL-TIME CLOCK*/
  OUTPUT(OFDH) = 20H; /*RESTORE INTELLEC INTERRUPT LOGIC*/
END CLOCK$3;

```

Figure 5-2. Formatted Clock Program

TEMPORARY SUSPENSION OF AUTOMATIC FORMATTING

It is also possible, when entering a program, do your own formatting on certain lines of code or to leave them unformatted. You can do this by using the comment `/*-f*/` to temporarily turn off the automatic formatting. The `/*-f*/` comment causes the lines of code below it to remain unformatted by the automatic formatting feature.

To restart the automatic formatting you would enter the comment `/*+f*/` on the line just before the lines of code you wish to be formatted with the automatic formatting feature.

Figures 5-3 and 5-4 show examples of the use of these special comments.

The comments shown in Figure 5-3 were entered in INSERT mode. They stopped automatic formatting on eight lines of code.

```

/* A CLOCK PGM: ACCEPTS TIME OF DAY IN HOURS, MINS, SECS */
/* THEN CONTINUOUSLY DISPLAYS TIME EVERY SECOND */
CLOCK:
DO;
/*-f*/
        DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0,0,0);
        DECLARE CLOCKSPULSES ADDRESS INITIAL (0);
        DECLARE LOCS8 BYTE AT (8);
        DECLARE LOCS9 ADDRESS AT (9);
        DECLARE BUFFER (100) BYTE;
        DECLARE BUFFPTR ADDRESS;
        DECLARE (COUNT, STATUS) ADDRESS;
        DECLARE CRLF (2) BYTE DATA (0DH, 0AH);

/*+f*/
DECLARE FOREVER LITERALLY 'WHILE 1';
$INCLUDE (:F1:INOUT.SRC)
$NOLIST
CLOCK$3:
PROCEDURE INTERRUPT 1;
  IF CLOCKSPULSES = 1024 THEN
    DO;
      CLOCKSPULSES = 0;
      SEC = SEC + 1;
      IF SEC = 60 THEN
        DO;
          SEC = 0;
          MIN = MIN + 1;
          IF MIN = 60 THEN
            DO;
              MIN = HOUR + 1;
              IF HOUR = 24 THEN
                DO;
                  HOUR = 0;
                END;
            END;
          END;
        END;
      END;
    END;
  END;
  OUTPUT(OFFH) = 2; /*ENABLE REAL-TIME CLOCK*/
  OUTPUT(OFDH) = 20H; /*RESTORE INTELLEC INTERRUPT LOGIC*/
END CLOCK$3;

```

Figure 5-3. Use of Special Comments

Figure 5-4 shows that the lines affected by the `/*-f*/` and `/*+f*/` comments can easily be changed. This is helpful if you change your mind about which lines you want to format yourself, for instance. In the example in Figure 5-4, the position of the `/*+f*/` comment was changed to affect an additional line of code.

```

/* A CLOCK PGM: ACCEPTS TIME OF DAY IN HOURS, MINS, SECS */
/* THEN CONTINUOUSLY DISPLAYS TIME EVERY SECOND */
CLOCK:
DO;
/*-f*/
        DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0,0,0);
        DECLARE CLOCKSPULSES ADDRESS INITIAL (0);
        DECLARE LOC88 BYTE AT (8);
        DECLARE LOC89 ADDRESS AT (9);
        DECLARE BUFFER (100) BYTE;
        DECLARE (COUNT, STATUS) ADDRESS;
        DECLARE CRLF (2) BYTE DATA (0DH, 0AH);
        DECLARE FOREVER LITERALLY 'WHILE 1';

/*+f*/

SINCLUDE (:F1:INOUT.SRC)
SNOLIST

CLOCK$3:
PROCEDURE INTERRUPT 1;
IF CLOCKSPULSES = 1024 THEN
DO;
    CLOCKSPULSES = 0;
    SEC = SEC + 1;
    IF SEC = 60 THEN
    DO;
        SEC = 0;
        MIN = MIN + 1;
        IF MIN = 60 THEN
        DO;
            MIN = HOUR + 1;
            IF HOUR = 24 THEN
            DO;
                HOUR = 0;
            END;
        END;
    END;
END;
END;
OUTPUT(OF8H) = 2; /*ENABLE REAL-TIME CLOCK*/
OUTPUT(OFDH) = 20H; /*RESTORE INTELLEC INTERRUPT LOGIC*/
END CLOCK$3;

```

Figure 5-4. Changing Lines Affected by Special Comments

HOW PSCAN WORKS

The PSCAN program consists of two modules: the text editor and the syntax checker/formatter. The text editing and syntax checking, and automatic formatting features have already been described. The following section describes how they work together.

During an editing session, the text editor's current position is the current cursor position. The syntax checker's current position is the point in the text up to which the text has been checked for syntax errors. Text that follows the current cursor position has not been checked for syntax.

In general, during PSCAN, the syntax checker lags behind the text editor. The syntax checker restarts checking for syntax errors from the very beginning of the program when one of the following occurs:

- o Text that has already been checked is changed.
- o Text before the point at which a syntax error error has been detected is changed.
- o The Set command is used to set the syntax checking or formatting on after it has been off earlier in the editing session.

The syntax checker is caught up with the editor when no syntax errors have been detected and its current position is not farther than a single line away from the current cursor position. If no message is displayed at the bottom of the screen you know that the syntax checker is caught up with the editor.

The syntax checker must always be caught up when the formatting option is on. This means that if the cursor is moved forward over a large number of lines (using the Jump command, for example) the syntax checker is not caught up, and PSCAN will not accept any more commands until the syntax checker catches up with the current cursor position.

When using PSCAN with large files (over 600 lines or 26 Kbytes), it is best to wait until the syntax checker is caught up with the text editor before entering more text. This can prevent delays between keystrokes because the checker may have to go to the disk to get the next line to be processed (large files require that some text be stored on disk).

PL/M-86 ORIENTED COMMANDS

To some extent, PSCAN's six PL/M-86 oriented editing commands depend on its syntax checking and automatic formatting features. This is because the PL/M-86 oriented editing commands are most effective on formatted text (which means that it must also have been checked for syntax).

Because PL/M-86 is a block-structured language, the commands included in PSCAN that allow you to move the cursor around within a PL/M-86 program according to program structure are especially helpful.

These special commands allow you to move the cursor to the beginning of the enclosed block in which the cursor is currently positioned, to move the cursor to the beginning of the first enclosed block, to move the cursor a statement up or a statement down (within the same indentation level), and to move the cursor a token to the left or to the right. Moving the cursor around using references to program structure can save time during editing of a PL/M-86 source file.

Table 5-1 lists the PL/M-86 oriented editing commands and their corresponding control characters. The control characters shown are the default control characters for these commands on a Series III. To know exactly what control characters to use on your terminal, check Appendix G which lists how several different terminals are configured.

Table 5-1. PL/M-86 Oriented Editing Commands

Pl/M-86 Oriented Editing Command	Control Character
Move-Level-Out	CONTROL Q
Move-Level-In	CONTROL W
Move-Statement-Up	CONTROL B
Move-Statement-Down	CONTROL N
Move-Token-Left	CONTROL O
Move-Token-Right	CONTROL P

To remember the control characters that invoke the PL/M-86 oriented editing commands on a Series III, note that commands that perform similar functions, for example Move-Level-In (CONTROL Q) and Move-Level-Out (CONTROL W) are invoked by keys that are positioned next to each other on the keyboard.

The Move-Level-Out command moves the cursor to the beginning of the enclosed block in which the cursor is currently positioned. If the cursor is not currently positioned in an enclosed block, the message "no enclosed block" appears in the message line of the display screen.

The Move-Level-In command moves the cursor to the beginning of the first enclosing block. If a block is enclosed by the block containing the cursor, either place the cursor at the beginning of that enclosed block or do not move the cursor. The message "no enclosed block" appears on the message line of the display screen in the latter case.

The Move-Statement-Up command moves the cursor to the statement immediately preceding the current cursor position (which must be at the same indentation level as the statement in which the cursor is currently positioned).

The Move-Statement-Down command moves the cursor to the statement immediately following the statement in which the cursor is currently positioned. (The statement to which the cursor moves must be at the same indentation level as the statement in which the cursor was positioned before the command was invoked.)

The Move-Token-Left command moves the cursor to the beginning of the syntactical token (or smallest meaningful unit) immediately preceding the current cursor position.

The Move-Token-Right command does the same thing as the Move-Token-Left command except that it moves the cursor to the beginning of the token immediately to the right of the current cursor position.

To invoke the Move-Statement-Down command you would press the control character that is configured to execute this command on your terminal (usually CONTROL N).

The cursor moves to the statement immediately following the statement in which it is currently positioned. The statement to which the cursor moves must be at the same indentation level as the one from which it moves. Figure 5-5 provides an example of the Move-Statement-Down command.

CURSOR
MOVES

```

/* A CLOCK PGM: ACCEPTS TIME OF DAY IN HOURS, MINS, SECS      */
/* THEN CONTINUOUSLY DISPLAYS TIME EVERY SECOND                */
/*                                                              */
CLOCK:
DO;
DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0,0,0);
DECLARE CLOCKSPULSES ADDRESS INITIAL (0);
DECLARE LOCSB BYTE AT (8);
DECLARE LOCS9 ADDRESS AT (9);
DECLARE BUFFER (100) BYTE;
DECLARE (COUNT, STATUS) ADDRESS;
DECLARE CRLF (2) BYTE DATA (ODH, OAH);
DECLARE FOREVER LITERALLY 'WHILE 1';

$INCLUDE (:F1:INOUT.SRC)
$NOLIST

CLOCK$1:
PROCEDURE INTERRUPT 1;
  IF CLOCKSPULSES = 1024 THEN
    DO;
      CLOCKSPULSES = 0;
      SEC = SEC + 1;
      IF SEC = 60 THEN
        DO;
          SEC = 0;
          MIN = MIN + 1;
          IF MIN = 60 THEN
            DO;
              MIN = HOUR + 1;
              IF HOUR = 24 THEN
                DO;
                  HOUR = 0;
                END;
            END;
          END;
        END;
      END;
    END;
  END;
  OUTPUT(OFFH) = 2; /*ENABLE REAL-TIME CLOCK*/
  OUTPUT(OPDH) = 20H; /*RESTORE INTELLEC INTERRUPT LOGIC*/
END CLOCK$1;

```

Figure 5-5. Move-Statement-Down Command

The PL/M-86 oriented editing commands help to decrease time spent editing statements or tokens and time spent making changes that affect blocks of program code.

In addition to the language oriented editing features described above, PSCAN has the ability to do limited semantic processing of LITERALLYs and \$INCLUDE files.

LITERALLYs are expanded correctly by both PL/M-86 and the PSCAN syntax checker.

PSCAN also processes all text appearing in files mentioned in THE \$INCLUDE compiler control statements of the text being edited in the primary buffer. If a program contains many \$INCLUDE compiler controls, PSCAN will take longer because the syntax checker must retrieve the content of the \$INCLUDE files for processing.



This appendix lists the PSCAN commands that appear on the menu prompt line of the display screen, their format, and a description of what they do when invoked.

The various special-purpose keys, function keys, and control characters used by default on a Series III or a Series IV terminal are listed in Table A-2 later in this Appendix. Most terminals have the same special-purpose keys with slight variations.

Table A-1 lists the PSCAN commands, their format and their description. Commands that cause mode transitions are indicated, and any subcommands that appear when certain commands are invoked are also noted in the table.

Table A-1. PSCAN Commands

Command	Format	Description
Again	[count] A	Repeats the last command or subcommand.
BLOCK (Mode)	B or D	Delimits section of text that can then be deleted, moved, or copied.
Buffer	B	Identifies the end of the section of text being delimited for copy to the BLOCK buffer.
Delete	D	Deletes the delimited section and moves it to the BLOCK buffer.
Find	F	Same as in main command level.
-find	-	Same as in main command level.
Jump	J	Moves cursor to location specified

		(by subcommand) in text.
Put	P	Copies text to named output file.
Delete	D or B	Delimits section of text that can then be deleted, moved, or copied (same as in BLOCK mode).
Execute	[count] E	Executes specified macro (must be invoked from main command level).
Get	[count] G	Retrieves contents of BLOCK buffer or external file; copies contents at current cursor position. Count must be a finite number.
Hex	[count] H	Hex command
Input	I	Assigns hexadecimal values to bytes at current cursor position.
Output	O	Displays hexadecimal values of ASCII characters at current cursor position in the message line.
INSERT (Mode)	I	Enters INSERT mode: inserts text at cursor.
Jump	J	Moves cursor to a specified location in text (according to subcommand).
End	E	Moves cursor to end of the file.
Line	L	Moves cursor to the start of the designated line.

Table A-1. PSCAN Commands (Cont'd.)

Position	P	Moves cursor to the designated column.
Start	S	Moves cursor to the start of the file.
A tag	A	Moves cursor to A tag.
B tag	B	Moves cursor to B tag.
C tag	C	Moves cursor to C tag.
D tag	D	Moves cursor to D tag.
Other	O	Switches between primary and secondary buffers.
QUIT (Mode)	Q	Ends editing session.
Abort	A	Returns to operating system; all changes are lost.
Exit	E	Returns to operating system; the file is updated.
Init	I	Restarts editing session - all changes are lost; initializes new file without returning to operating system.
Update	U	Updates file without returning to operating system.
Write	W	Writes file to the specified output file without returning to operating system.
Replace	[count] R	Searches forward for target string; replaces it with new string if found.

Table A-1. PSCAN Commands (Cont'd.)

?replace	[count] ?	Conditional Replace command.
Set	S	Sets several PSCAN features, e.g., Autocr, tabs. Controls several yes/no options for editing environment.
Autocr	A	While in INSERT mode, inserts carriage return in text automatically when line is full. (Default = no)
BAK file	B	Creates backup file of file you are currently editing before QUIT Update or QUIT Exit command replaces it. (Default = yes)
Case	C	Tells PSCAN to consider case of strings during -find and Replace commands. (Default = no)
Err_check	E	Sets syntax checking feature on or off. (Default = on)
Format	F	Sets automatic formatting feature on or off. (Default = on)
Indent	I	Sets automatic indenting. (Program text lines are indented using tabs to reflect program structure.) (Default = off)

Table A-1 PSCAN Commands (Cont'd.)

Leftcol	L	Sets left column. (Default = 0).
Notab	N	Inserts blanks in place of tabs. (Default = no).
Showfind	S	Lists lines of multiple finds. (Default = no)
Tabs	T	Sets tab positions.
Viewrow	V	Sets row to which cursor moves on View command. (Default = row 5)
Tag	T	Specifies maximum of four locations in a file.
View	V	Rewrites screen leaving cursor in viewrow. (Default = row 5)
XCHANGE (Mode)	X	Enters XCHANGE mode: replaces characters on a one-for-one basis.
Macro Commands		
MACRO (Mode)	M	Enters MACRO mode. Can create macros, retrieve them from memory, and insert them in text.
Create	C	Creates macros, retrieves them from memory and inserts them in text.
Get	G	Retrieves macros from an external file or from the text buffer.

Table A-1 PSCAN Commands (Cont'd.)

Insert	I	Inserts subsequent input in text in macro form.
List	L	Lists names of all currently defined macros in message line.
Save	S	Saves macros in form suitable for macro files.

SPECIAL-PURPOSE KEYS AND CONTROL CHARACTERS

Table A-2 lists special-purpose keys and control characters that, when pressed, invoke certain PSCAN commands. Note that the keys themselves do not perform the functions indicated. When pressed, each key or control character generates a unique code that is interpreted by the PSCAN program in order to perform a specific function.

Table A-2 shows the special-purpose and control characters that are found on most terminals. The functions these keys/control characters have been configured to perform are those that they are usually configured to perform. However, these functions can be changed by using the configuration commands described in Appendix G.

Table A-2. PSCAN's Use of
Special-Purpose Keys
and Control Characters

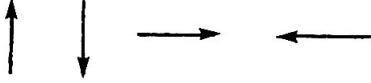
Special-purpose Key	Function
Directional Arrows 	Moves the cursor in the direction of the arrow.
<ESC>	Returns PSCAN to main command level.

Table A-2. PSCAN's Use of
Special-Purpose Keys
and Control Characters (Cont'd.)

<RETURN>	Places the cursor at the beginning of the next line of text.
<TAB>	Causes the next line of available commands to be displayed in the menu prompt line of the display screen.
<RUBOUT>	Deletes the character that precedes the cursor. The cursor moves back one, taking the place of the deleted character.
Control Character	Function
CONTROL A	Deletes the character under the cursor and the characters to the end of the line.
CONTROL B	Executes Move-Statement-Up command.
CONTROL C	Aborts any command and returns PSCAN to main command level.
CONTROL D	Traps to the debugger.
CONTROL F	Deletes character under the cursor.
CONTROL N	Executes Move-Statement-Down command.

Table A-2. PSCAN's Use of
Special-Purpose Keys
and Control Characters (Cont'd.)

CONTROL O	Executes Move-Token-Left command.
CONTROL P	Executes Move-Token-Right command.
CONTROL Q	Executes Move-Level-Out command.
CONTROL T	Executes Move-Level-In command.
CONTROL U	Restores characters deleted by previous delete command.
CONTROL X	Deletes the characters to the left of the cursor
CONTROL Y	Executes Move-Level-Out Command
CONTROL Z	Deletes line in which the cursor is currently positioned

The following special-purpose keys appear on the Series IV keyboard in addition to those listed above:

<CAPS LOCK>	Controls the case of letter characters.
<CHAR DELETE>	Deletes the character currently under the cursor.
<CLEAR LINE>	Deletes the line of text in which the cursor is currently positioned.

FUNCTION KEYS

Function keys labeled F0 - F7 also appear on the Series IV keyboard. They are configurable; they are used by default to invoke the command displayed on the screen above the function key (they change from screen to screen, depending on what commands are currently displayed).

Appendix E includes a picture of the Series IV keyboard in which the function keys can be seen.



.

.



.

.



APPENDIX B
ERROR MESSAGES

This appendix lists the error messages reported by PSCAN when a problem is encountered during a command. Each message is printed on one line, followed by an explanation of the message following it.

EDITING COMMAND ERRORS

bad tabs

Attempt to set bad tabs: e.g., 4, 2. PSCAN returns to main command level.

bad Leftcol

Attempt to set bad left column: e.g., PSCAN returns to main command level.

bad Viewrow

Attempt to set bad viewrow: e.g., 45. PSCAN returns to main command level.

cannot delete more than 32

Attempt to use count greater than 32 with <DELETE CHAR> (or equivalent control character). PSCAN returns to main command level.

excep# or excep: filename line#

UDI exception number was returned. The second message shown appears if the exception occurs in an INCLUDE file.

illegal command

Attempt to enter illegal command. PSCAN ignores command.

invalid hex value

Attempt to input an invalid hex value. PSCAN returns to main command level.

line too long or line too long: filename line #

The line is too long (i.e., greater than 180 characters). The second message shown appears if the line is in an INCLUDE file.

nesting

The nesting of the INCLUDE files is too deep. Nesting cannot be greater than 9 files deep.

no errors exist

No syntax errors have been detected by the syntax error checker.

no such macro

Attempt to execute macro that does not exist. PSCAN returns to main command level.

no enclosed block

The block in which the cursor is currently positioned does not enclose any other block.

no enclosing block

No block encloses the block in which the cursor is currently positioned.

no preceding token

No token precedes the current cursor position.

no succeeding token

No token follows the current cursor position.

no preceding statement

No statement of the same indentation level precedes the current cursor position.

no succeeding statement

No statement of the same indentation level follows the current cursor position.

not found: "target string"

Target string not found. PSCAN returns to main command level.

some text lost

Attempt to increase contents of secondary buffer past 7 Kbytes. Output file changed to :BB: to avoid inadvertent loss of file with a QUIT Exit command.

syntax error or syntax error: filename line #

Syntax error checker has detected a syntax error in the structure of the language. If the error occurs in an INCLUDE file, the second message appears

filename <error message supplied by operating system>

An error occurs during a QUIT Exit, QUIT Update, Get, or BLOCK Put command. PSCAN returns to main command level.

Xchange limit is 100

Attempt to replace over 100 characters without restarting XCHANGE mode. PSCAN remains in XCHANGE mode.

MACRO FILE ERRORS

If any error is found in a macro file, one of the following messages is printed (where nnn is the line of the macro file that contains the error) and macro file processing continues. If more than one error is found, PSCAN appends the line that follows:

; error in nnn[, nnn...]

to the message that explains the first error encountered, where each nnn is the line number of an additional error.

error in nnn no more room for macros

Attempt to create a macro when macro storage is full. Macro definition is terminated.

error in nnn bad hex value

Configuration command contains bad hex value, e.g., 3G.

error in nnn bad '/' code

Backslash (\) is not followed by a valid value.

error in nnn missing ';'

Set command is not terminated with a semicolon or a carriage return.

error in nnn missing '='

error in nnn bad AF type
bad AV value
bad AX value

error in nnn bad command

Macro definition contains a bad command; unknown control code or character, i.e., not M or S.

error in nnn no macro name

Macro definition does not include macro name.

INVOCATION ERRORS

Insufficient Memory

PSCAN does not have a large enough RAM partition. PSCAN exits to operating system.

Illegal Invocation Line

Invocation line contains an unknown option. This error is not final (i.e., PSCAN does not exit to operating system level).

Table C-1. ASCII Codes

ASCII Character	Control Character	HEX	ASCII Character	HEX
NUL		00	@	40
SOH	CONTROL A	01	A	41
STX	CONTROL B	02	B	42
ETX	CONTROL C	03	C	43
EOT	CONTROL D	04	D	44
ENQ	CONTROL E	05	E	45
ACK	CONTROL F	06	F	46
BEL	CONTROL G	07	G	47
BS	CONTROL H	08	H	48
HT	CONTROL I	09	I	49
LF	CONTROL J	0A	J	4A
VT	CONTROL K	0B	K	4B
FF	CONTROL L	0C	L	4C
CR	CONTROL M	0D	M	4D
SO	CONTROL N	0E	N	4E
SI	CONTROL O	0F	O	4F
DLE	CONTROL P	10	P	50
DC1	CONTROL Q	11	Q	51
DC2	CONTROL R	12	R	52
DC3	CONTROL S	13	S	53
DC4	CONTROL T	14	T	54
NAK	CONTROL U	15	U	55
SYN	CONTROL V	16	V	56
ETB	CONTROL W	17	W	57
CAN	CONTROL X	18	X	58
EM	CONTROL Y	19	Y	59
SUB	CONTROL Z	1A	Z	5A
ESC		1B	[5B
FS		1C	\	5C
GS		1D]	5D
RS		1E	^(↑)	5E
US		1F	~	5F
space		20		60
!		21	a	61
"		22	b	62
#		23	c	63
\$		24	d	64
%		25	e	65
&		26	f	66
'		27	g	67
(28	h	68
)		29	i	69
*		2A	j	6A
+		2B	k	6B
,		2C	l	6C
-		2D	m	6D
.		2E	n	6E
/		2F	o	6F
0		30	p	70
1		31	q	71
2		32	r	72
3		33	s	73
4		34	t	74
5		35	u	75
6		36	v	76
7		37	w	77
8		38	x	78
9		39	y	79
:		3A	z	7A
;		3B	{	7B
<		3C		7C
=		3D	}	7D
>		3E	~	7E
?		3F	DEL	7F



PSCAN runs on the Intel Series III Microcomputer Development System under the ISIS operating system.

MEMORY

The PL/M-86 syntax checking editor requires a configuration of 160 Kbytes of RAM.

INVOCATION COMMAND

The invocation, which must be preceded by RUN, is as follows:

```
RUN PSCAN [input file[TO output file]] [,other input file[TO other  
output file]]  
[{MACRO, MR} [(macro file)]|{NOMACRO, NOMRR}]
```

Upon invocation the screen displays the following:

```
----Series III PSCAN Vx.y, Copyright Intel Corporation, 1984  
Again Block Delete Execute Find -find Get --more--
```

KEYBOARD

On the Series III, commands are invoked by pressing the first character of the command displayed on the menu prompt line of the display screen, or by pressing the special-purpose key or control character that performs the desired command.

The TPWR key provides lowercase entry (latched position) or uppercase entry (unlatched position) of alphabetic characters.



.

.



.

.



PSCAN runs on the Series IV Microcomputer Development System under the iNDX operating system. Because PSCAN is a 16-bit program, it cannot be invoked under the ISIS operating system on the Series IV.

MEMORY

The PL/M-86 syntax checking editor requires a configuration of 160 Kbytes of RAM.

INVOCATION COMMAND

The invocation is as follows:

```
PSCAN [input file[TO output file]] [,other input file[TO other  
output file]]  
[{MACRO, MR} [(macro file)] [{NOMACRO, NOMR}]
```

Upon invocation, the screen displays the following:

```
----Series IV PSCAN, Vx.y, Copyright Intel Corporation, 1984  
Again Block Delete Execute Find -find Get -more-
```

KEYBOARD

In addition to its other keys, the Series IV keyboard contains eight function keys labeled F0 - F7. These function keys perform the commands displayed on the menu prompt line under which they appear on the keyboard. The commands the function keys execute vary depending on what commands are displayed in the menu prompt line.

To invoke a command with the Series IV keyboard, press the initial character of the desired command displayed on the screen, the function key that appears under the desired command, or the appropriate control character or special-purpose key. (The special-purpose keys generally control cursor movement and deletion.)

The Series IV also contains the following special-purpose keys, in addition to the special-purpose keys described for the Series III.

CHAR DELETE deletes the character over which the cursor is positioned.

CLEAR LINE deletes the entire line in which the cursor is positioned.

CAPS LOCK provides lowercase entry (latched position) or uppercase entry (unlatched position) of alphabetic characters.

The Series IV keyboard is shown in Figure E-1.

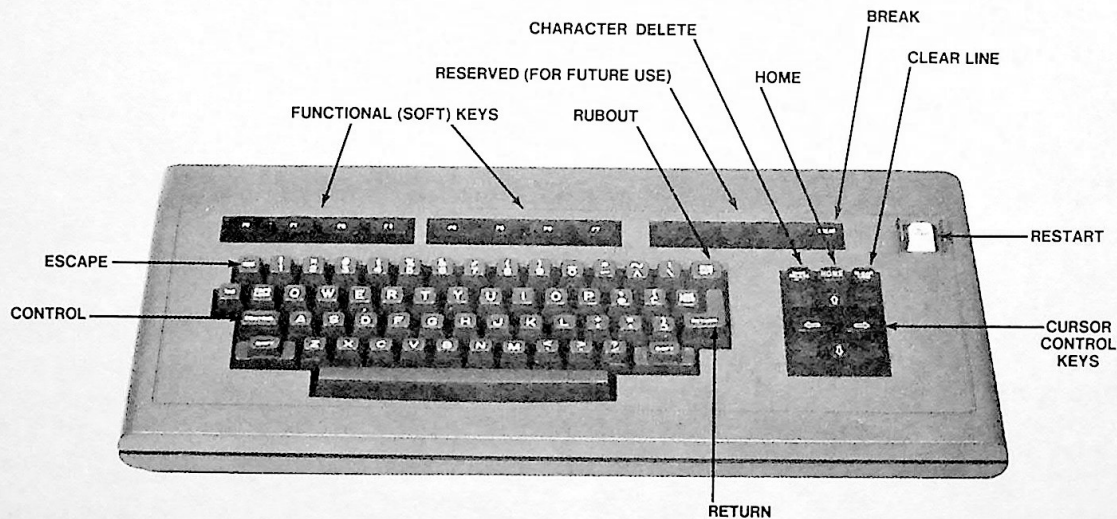


Figure E-1. Series IV Keyboard

MEMORY

PSCAN requires a configuration of 160 Kbytes of RAM.

INVOCATION COMMAND

The invocation command under iRMX is as follows:

```
PSCAN [input file [To output file] [, other input file [To other  
output file]]  
[{MACRO,MR} [(macro file)] | {NOMACRO,NOMR}]
```

Upon invocation, the screen displays the following:

```
----system id PSCAN Vx.y, Copyright Intel Corporation, 1984
```

```
Again  Block  Delete  Execute  Find  -find  Get      --more--
```



During a PSCAN session, the PSCAN program executes in the computer and communicates with the user via the terminal.

The terminal consists of a CRT screen and a keyboard. The keyboard contains many keys, each of which, when pressed, generates a unique number (code) that is then transmitted to the computer to which the terminal is attached. The exact code is decided upon by the manufacturer of the terminal. Similarly labeled keys on different terminals may generate different codes.

The CRT screen contains a cursor that can be moved around on the screen. The screen has the ability to display characters and to insert and delete characters at the current cursor position. Both the cursor and the screen are controlled by codes sent to the screen by the program that is executing in the computer.

In order for the keyboard to perform its input functions, the codes generated by the keyboard must be meaningful to PSCAN. In other words, PSCAN's input commands must be mapped to the codes generated by the various keys on the terminal's keyboard.

Similarly, to use the screen correctly, PSCAN must be aware of the codes that will perform a certain screen/cursor function for the particular terminal. The terminal's screen codes must be mapped to PSCAN's output screen commands.

CONFIGURATION COMMANDS

The mapping of a terminal's keyboard and screen codes to PSCAN's keyboard and screen commands is done using PSCAN's configuration commands. These configuration commands must be entered in the form:

command = code;

The command is a name for one of PSCAN's screen or keyboard commands. The code is a screen or keyboard code used by the terminal. The = sign maps the code to the command and indicates to PSCAN that the "code" represents the PSCAN "command" whenever PSCAN is used with this particular terminal.

Example 1

AFCD = 16;

AFCD is the name given to PSCAN's keyboard input command Move Cursor Down. This configuration command tells PSCAN that the user is asking PSCAN to move the cursor down on the screen whenever PSCAN receives code number 16 from the user's keyboard.

Example 2

AFND = 16;

The command AFND is the name of PSCAN's output screen command Move Cursor down One Line. This configuration command tells PSCAN to send code number 16 to the screen whenever it wants the screen electronics to move the cursor down one line.

On a Televideo 925, for example, the key that generates code number 16 is labeled ↓. A code of 16 sent to the screen by the executing program causes the cursor to move down one line.

CONFIGURING PSCAN

To configure PSCAN to run on a specific terminal, you must create a file containing all of PSCAN's configuration commands. For each command, you must have the correct code that is to be used by the terminal. In addition to configuration commands, this file can also contain Set commands and MACRO definitions. PSCAN must read this file upon invocation so that it can read the configuration commands and use the configuration information to run correctly on the particular terminal. There are several ways to make this happen:

- o Name the file PSCAN.MAC and place it in the same directory as PSCAN. Do not include a macro control in the invocation line. When PSCAN is invoked, it will look in its directory for a file named PSCAN.MAC, from which it will read the configuration information and set up commands and MACROS.
- o Name the file as you wish and use the MACRO (filename) control in the invocation line. When PSCAN is invoked, it will not look for PSCAN.MAC. It will get its configuration information from the file that was named in the invocation line with the MACRO control.

- o After invoking PSCAN, do a MACRO Get command from main command level, "getting" the file that contains the configuration commands. PSCAN will use the configuration commands to reconfigure itself while it is running.

Warning

Any Set commands and MACROS that exist in the file which you "get" will also be used by PSCAN. This could cause redefinition of some MACROS that have names that are the same as those contained in the file that is read in with the Get command.

PSCAN does not need to be configured in order to run on a Series III or a Series IV; it will configure itself to run on either of these terminals. However, if, when using a Series III or Series IV, a PSCAN.MAC describing a different terminal exists in the same directory as PSCAN, you must indicate to PSCAN in the invocation line not to read PSCAN.MAC. This is done by using the NOMACRO control in the invocation line, for example:

```
RUN PSCAN MODULE.01 NOMACRO
```

Table G-1 lists the terminal codes used by several terminals that can be mapped to PSCAN's configuration commands to enable PSCAN to run on these particular terminals.

Table G-1. Configuration Commands

Macro Filename	.MAC	.MAC	L151?.MAC		LADM3A MAC	LT910P. MAC	LT925. MAC	LVT52. MAC	LVT100. MAC	
Command Input	Series-IV Default	Series-III Default	Hazeltine		Lear Sig ADM3A	Televideo 910'	Televideo 925	VT 52	VT 100	Meaning
			E 1510	T Tilde						
AV =	25;	25;	24;	24;	24;	24;	24;	24;	24;	Sets the number of lines of the display.
AB =	1BH;	1BH;	7E;		1B;	1B;	1B;	0B;	0B;	Sets ESC.
AR =	7FH;	7FH;	7F;	7F;	7F;	7F;	7F;	7F;	7F;	Sets RUBOUT
AFXA =	1H;	1H;	01;	01;	01;	01;	01;	01;	01;	Sets DELETE RIGHT. CONTROL A.
AFXF =	80H;	6H;	06;	06;	06;	06;	06;	06;	06;	Sets CHAR DELETE. CONTROL F.
AFXU =	15H;	15H;	15;	15;	15;	15;	15;	15;	15;	Sets UNDO. CONTROL U.
AFXX =	18H;	18H;	18;	18;	18;	18;	18;	18;	18;	Sets DELETE LEFT. CONTROL X.
AFXZ =	82H;	1AH;	1A;	1A;	1A;	1A;	1A;	1A;	1A;	Sets CLEAR LINE. CONTROL Z.
AFCD =	88H;	1CH;	0B;	0B;	CA;	16;	16;	1B42;	1B42;	Sets DOWN.
AFCH =	81H;	1DH;	12;	12;	1E;	1E;	1E;	0F;	0F;	Sets HOME.
AFCL =	89H;	1FH;	08;	08;	08;	08;	08;	1B44;	1B44;	Sets LEFT.
AFCR =	8AH;	14H;	10;	10;	0C;	0C;	0C;	1B43;	1B43;	Sets RIGHT.
AFCU =	87H;	1EH;	0C;	0C;	0B;	0B;	0B;	1B41;	1B41;	Sets UP.
AFIG =			1B;	7E;	;	;	;	;	;	This character will be ignored if input. This character is needed on terminals (such as the Hazeltine 1510) which have multiple character key codes for UP and DOWN. AFIG should be set to the lead in (tilde) and UP and DOWN should be set to the second letter of the cursor up or down key code. This avoids problems caused by the lack of a typeahead buffer.
AFLO =	14H;	14H;	11;	14;	14;	14;	14;	14;	14;	Moves cursor level out. CONTROL Q.
AFLI =	19H;	19H;	17;	19;	19;	19;	19;	19;	19;	Moves cursor level in. CONTROL W.
AFSU =	02H;	02H;	16;	16;	02;	02;	02;	02;	02;	Moves cursor statement in. CONTROL B.
AFSD =	0EH;	0EH;	02;	02;	0E;	0E;	0E;	0E;	0E;	Moves cursor statement down. CONTROL N.
AFTL =	0FH;	0FH;	09;	09;	0F;	0F;	0F;	11;	11;	Moves cursor token left. CONTROL O.
AFTR =	10H;	10H;	0F;	0F;	10;	10;	10;	17;	17;	Moves cursor token right. CONTROL P.

Table G-1. Configuration Commands (Cont'd.)

Macro Filename	.MAC	.MAC	L151?.MAC		LADM3A MAC	LT910P. MAC	LT925. MAC	LVT52. MAC	LVT100. MAC	
Command Output	Series-IV Default	Series-III Default	Hazeltine		Lear Sig ADM3A	Teletideo 910'	Teletideo 925	VT 52	VT 100	Meaning
			E	T						
			1510	Tilde						
AFMB =	0DH;	0DH;	0D;	0D;	0D;	0D;	0D;	1B42;	0D;	Moves cursor to start of line.
AFMD =	1B02H;	1CH;	1B0B;	7E0B;	0A;	16;	16;	1B48;	1B42;	Moves cursor down.
AFMH =	1B08H;	1DH;	1B12;	7E12;	1E;	1E;	1E;	1B44;	1B48;	Moves cursor home.
AFML =	1B04H;	1FH;	08;	08;	08;	08;	08;	1B43;	1B44;	Moves cursor left.
AFMR =	1B03H;	14H;	10;	10;	0C;	0C;	0C;	1B41;	1B43;	Moves cursor right.
AFMU =	1B01H;	1EH;	1B0C;	7E0C;	0B;	0B;	0B;		1B41;	Moves cursor up.
AFES =	1B05H;	1B1CH;	7E1C;	7E1C;	1A;	1B2B;	1B2B;	;	;	Erases entire screen.
AFER =	1B10H;	1B4AH;	1B1B;	7E18;	;	1B59;	1B59;	1B4A;	1B4A;	Erases rest of screen.
AFEK =		1B4BH;	;	;	;	;	;	;	;	Erases entire line.
AFEL =	1B11H;		1B0F;	7E0F;	1B54;	1B54;	1B54;	1B4B;	1B4B;	Erases rest of line.
AFAC =	1B19H;		1B11;	7E11;	1B3D;	1B3D;	1B3D;	1B59;	1B59;	Addresses cursor lead in. When used, code will be followed by column number (0 to 79) and row number (0 to 24).
AO =	20H;	0H;	0;	0;	20;	20;	20;	20;	20;	Offset to add both row and column number with address cursor command.
AX =	F;	T;	F;	F;	F;	F;	F;	F;	F;	True if X (column) precedes Y (row) in address cursor command.
AW =	F;	T;	F;	F;	F;	F;	F;	F;	F;	Allows user to indicate that the terminal wraps when a character is printed in column 80.
AFIL =			1B1A;	7E1A;	;	1B45;	1B45;	1B49;	1B49;	Inserts line code. Used in line 0 for reverse scrolling.
AFDL =			1B13;	7E13;	;	1B52;	1B52;	;	1B52;	Deletes line code. Used to speed up display on the Hazeltine 1510 and similar terminals.
AFBK =	20H;	20H;	20;	20;	20;	20;	20;	20;	20;	Blankout character. <BLANK> on most terminals.



.

.



.

.



PSCAN can be used noninteractively as well as interactively. During noninteractive use, PSCAN receives command input from a file (instead of from the keyboard), sends output to a file (instead of to the display screen), or both.

NONINTERACTIVE INPUT

You can send commands to PSCAN from a file by placing the commands in a SUBMIT (or similar batch program) file and then running the SUBMIT program.

You can place most commands in the file just as you would enter them from the keyboard: in other words, enter the command's initial letter. However, you must use the Hex Input command to enter nonprintable command characters and codes for ESC, CONTROL C, delete commands, and cursor movement commands.

For example, if you want to create a SUBMIT file that replaces all occurrences of the string CLOCK with TIMER in the CLOCK program using PSCAN, create a SUBMIT file (a file with extension .CSC) that contains the following:

```
[RUN] PSCAN %0
SSN SCN /RCLOCK <ESC> TIMER <ESC> QE
```

Use INSERT mode to enter all the commands except ESC. To insert the ESC commands, use Hex Input. Question marks (?) will appear on your screen where you insert hex value 1B, the value for ESC (on the Series III).

The SUBMIT file thus contains the following commands:

Command	Function
[RUN] PSCAN %0	Invocation Line
SSN	Set Showfind No (turns off Showfind option)
SCN	Set Case No (turns off Case option)
/	Forever Count
R	Replace command
CLOCK	The words to replace
?	appears in place of 1B hex, or ESC

Command (Cont'd.)	Function (Cont'd.)
TIMER	The words to put in the place of CLOCK
?	appears in place of 1B hex, or ESC
Q	QUIT
E	Exit

If you insert a carriage return/line feed (with the RETURN key) in the SUBMIT file, PSCAN will interpret the line feed as part of the command input.

To ignore the carriage return/line feed sequence, create a macro file named Config containing the following configuration command:

```
AFIG=ODOA;
```

Terminate the command with a semicolon (;). To cause the PSCAN to read the macro file Config, place the command MACRO Get and the macro file name "Config" after the invocation line in the SUBMIT file.

NONINTERACTIVE OUTPUT

You can use the CONSOL command (included in the ISIS-II Software Toolbox) to redirect PSCAN screen output to another logical device such as a file.

For example, the command-

```
CONSOL ,:F1:OUTFIL
```

directs output to the file OUTFIL on device :F1:.

On a Series III system, enter the following in the SUBMIT file:

```
CONSOL ,:F1:OUTFIL  
RUN PSCAN...
```

This will direct output to the file OUTFIL on device :F1:.

This appendix contains a printed version of the CLOCK program that is used in the tutorial in Chapter 2 and in examples throughout the manual.

This printout is of the unformatted program.

```
/* A CLOCK PGM: ACCEPTS TIME OF DAY IN HOURS, MINS, SECS      */
/* THEN CONTINUOUSLY DISPLAYS TIME EVERY SECOND                */
```

```
CLOCK:
DO;
DECLARE (HOUR, MIN, SEC) BYTE INITIAL (0, 0, 0);
DECLARE CLOCK$PULSES ADDRESS INITIAL (0);
DECLARE LOC$8 BYTE AT (8);
DECLARE LOC$9 ADDRESS AT (9)
DECLARE BUFFER(100) BYTE;
DECLARE BUFFPTR ADDRESS;
DECLARE (COUNT, STATUS) ADDRESS;
DECLARE CRLF (2) BYTE DATA (ODH, OAH);
DECLARE FOREVER LITERALLY 'WHILE 1';
```

```
$INCLUDE (:F1:INOUT.SRC)
$NOLIST
```

```
CLOCK$3:
PROCEDURE INTERRUPT 1;
IF CLOCK$PULSES = 1024 THEN
DO;
CLOCK$PULSES = 0;
SEC = SEC + 1;
IF SEC = 60 THEN
DO;
SEC = 0;
MIN = MIN + 1;
IF MIN = 60 THEN
DO;
IF HOUR = 24 THEN
DO;
HOUR = 0;
END;
END;
END;
END;
OUTPUT(OFFH) = 2;
OUTPUT(OFDH) = 20H;
END CLOCK$3;
```

```
/*ENABLE REAL-TIME CLOCK*/
/*RESTORE INTELLEC INTERRUPT LOGIC*/
```



ASCII	The American Standard Code for Information Interchange.
block	A unit of program structure that may be considered complete, such as the entire sequence of statements contained in a compound statement.
BLOCK mode	The mode used to delimit a block of text that then can be deleted, saved, or placed in another file.
character	The smallest unit of program text, such as the letter a, or the delimiter ;.
command	A direction given to the editor specifying a particular action the editor is to perform. To invoke a command either press the named letter key or a combination of the letter key and the CONTROL key.
compiler control	A directive to the compiler embedded in the text of a program. While not a part of the program itself, the compiler control is indicated by a dollar sign character (\$) at the beginning of a line in the program text.
extension	PSCAN presumes that a file name (or path name) may contain a three-character sequence that specifies the type of file. For example, the extension OBJ commonly is interpreted to mean the file is an object file. Similarly, the extension P86 may be used to indicate that the file is a PL/M-86 source file.
format	The method by which PSCAN arranges text of a program source file to conform to the structure of the language. To format, PSCAN uses line breaks and indentation levels.
full pathname	The complete specifier used to access a file.
indentation level	The number of blank columns preceding the first non-blank character on a line of program text.
INSERT mode	The mode used to enter the program text.

invocation control	A direction to PSCAN at the time it is invoked.
language module	A file containing the information necessary for LOE to perform syntax checking and program formatting.
main command level	The mode in which PSCAN comes up when it is invoked. All other modes must be entered from main command level, except for MACRO mode, which can be entered from any mode.
line break	The carriage return line feed combination that breaks the text of a program into distinct lines.
MACRO mode	The mode from which a macro is created, listed, saved, or retrieved.
mode	PSCAN operates in six modes. A series of commands, and in some cases subcommands, is available in each of the modes.
QUIT mode	Allows exit of the program or the editor.
structure	The hierarchical organization of a program in terms of statements, declarations, etc. reflecting the dynamic behavior of the program when executing.
source file	A PL/M-86 program that will require translation to machine program language before use.
statement	The unit of program structure that specifies the actions to be performed during program execution. A statement may be compound: i.e., it may contain more than one statement.
syntactic correctness	The property of a complete program or of a statement that ensures that it obeys all the rules of the grammar of the language in which it is written.
token	A group of characters in a program source file that make up a single logical unit. For example, in an assignment statement "abc = 20+3," the abc is the identifier token and 20 is a constant token.
whitespace character	Characters such as carriage return, line feed, and tab and control characters. These characters do not have a printed representation.
XCHANGE mode	Permits replacing text without first deleting it.

Again Command, 3-15, A-1
ASCII Codes, C-1 thru C-2
Automatic formatting, 1-1, 5-3
Available Commands
Within Modes, 3-24
BLOCK Mode Commands, 3-10, A-1
 Buffer, 3-11, A-1
 Delete, 3-11, A-1
 Put, 3-11, A-2
Comment,
 in Macro Files, 4-2
 special comments, 5-4
Configuration Commands, G-1, G-4
Configuring PSCAN, G-2
Control Characters, 2-2 thru 2-3, A-7
Delete Command, 3-12, A-2
Deleting Macros, 4-18
Editing Command Errors, B-1 thru B-3
Execute Command, 4-17, A-2
Find, -find command, 3-6, A-1
Function Keys, 2-2 thru 2-3, A-8
Get Command, 3-13, A-2
Hex Command, 3-15, A-2
INCLUDE files, 5-9
Insert Command, 3-7, A-2
Invocation
 Errors, B-4
 Line, 2-5
Jump Command, 3-9, A-2
LITERALLYs, 1-1, 5-9
MACRO commands, 4-11
 Create, 4-11 thru 4-12, A-5
 Get, 4-13, A-5
 Insert, 4-14, A-5
 List, 4-14, A-5
 Save, 4-15, A-6
MACRO
 Creation, 4-1
 Definition, 4-8
 Examples, 4-5 thru 4-8
 Files, 4-2
 File errors, B-3
 Form symbols, 4-9
 Names, 4-2
MODES, 3-1
 Main Command Level, 3-1
 INSERT, 3-1, 3-7
 XCHANGE, 3-1, 3-17

- QUIT, 3-1, 3-21
- BLOCK, 3-1, 3-10
- MACRO, 3-1, 4-1
- Noninteractive
 - Input, H-1
 - Output, H-2
- Other Command, 3-13, A-3
- PL/M-86 Oriented Commands, 3-17 thru 3-20
 - Move-Level-Out, 3-18, 5-7
 - Move-Level-In, 3-18, 5-7
 - Move-Statement-Up, 3-19, 5-7
 - Move-Statement-Down, 3-19, 5-7
 - Move-Token-Left, 3-19, 5-7
 - Move-Token-Right, 3-20, 5-7
- QUIT Mode Commands, 3-20, A-3
 - Abort, 3-21, A-3
 - Exit, 3-21, A-3
 - Init, 3-21, A-3
 - Update, 3-22, A-3
 - Write, 3-22, A-3
- Replace, ?replace,
Command, 3-8, A-3
- Set Command, 3-2 thru 3-6, A-4
 - Autocr, 3-3
 - BAK, 3-3
 - Case, 3-3
 - Err_check, 3-4
 - Format, 3-4
 - Indent, 3-4
 - Leftcol, 3-4
 - Notab, 3-5
 - Showfind, 3-5
 - Tabs, 3-5
 - Viewrow, 3-6
- Special-purpose keys, 2-2 thru 2-5, A-6
- Syntax Checking, 1-1, 5-1
- Tag Command, 3-14, A-5
- Undo Command, 3-14 thru 3-15
- View Command, 3-15, A-5
- Xchange Command, 3-17, A-3



REQUEST FOR READER'S COMMENTS

Intel's Technical Publications Departments attempt to provide publications that meet the needs of all Intel product users. This form lets you participate directly in the publication process. Your comments will help us correct and improve our publications. Please take a few minutes to respond.

Please restrict your comments to the usability, accuracy, readability, organization, and completeness of this publication. If you have any comments on the product that this publication describes, please contact your Intel representative. If you wish to order publications, contact the Intel Literature Department (see page ii of this manual).

1. Please describe any errors you found in this publication (include page number).

2. Does the publication cover the information you expected or required? Please make suggestions for improvement.

3. Is this the right type of publication for your needs? Is it at the right level? What other types of publications are needed?

4. Did you have any difficulty understanding descriptions or wording? Where?

5. Please rate this publication on a scale of 1 to 5 (5 being the best rating).

NAME _____ DATE _____

TITLE _____

COMPANY NAME/DEPARTMENT _____

ADDRESS _____

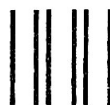
CITY _____ STATE _____ ZIP CODE _____

(COUNTRY)

Please check here if you require a written reply. ☐

WE'D LIKE YOUR COMMENTS ...

This document is one of a series describing Intel products. Your comments on the back of this form will help us produce better manuals. Each reply will be carefully reviewed by the responsible person. All comments and suggestions become the property of Intel Corporation.



NO POSTAGE
NECESSARY
IF MAILED
IN U.S.A.

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 1040 SANTA CLARA, CA

POSTAGE WILL BE PAID BY ADDRESSEE

Intel Corporation
Attn: Technical Publications M/S 6-2000
3065 Bowers Avenue
Santa Clara, CA 95051





.

.



.

.





INTEL CORPORATION, 3065 Bowers Avenue, Santa Clara, California 95051 (408) 987-8080

Printed in U.S.A.